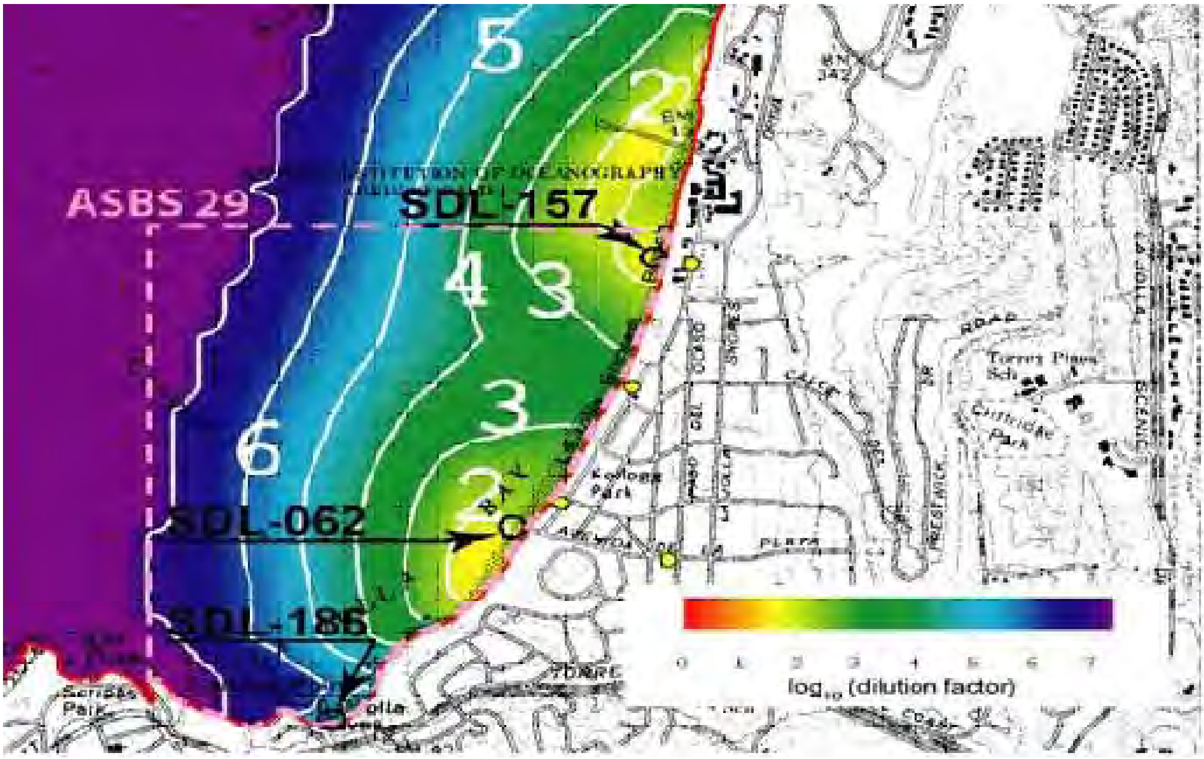


City of San Diego
La Jolla Area of Special Biological Significance
Site Specific Dilution and Dispersion Model

June 2013

City of San Diego

Prepared by:
AMEC Environment and Infrastructure, Inc.
Project No. 5025121039

LA JOLLA AREA OF SPECIAL BIOLOGICAL SIGNIFICANCE
SITE SPECIFIC DILUTION AND DISPERSION MODEL

Prepared for:
City of San Diego
Transportation and Storm Water Department
9370 Chesapeake Drive, Suite 100
San Diego, California 92123

Submitted by:
AMEC Environment & Infrastructure, Inc.
9210 Sky Park Court, Suite 200
San Diego, California 92123

May 2013

AMEC Project No. 5025121039

This special study report is prepared for the City of San Diego, Transportation and Storm Water Department (City of San Diego) to complement multiple, ongoing storm water monitoring programs within the La Jolla Area of Special Biological Significance, No 29, located in La Jolla, California. In particular, this study is designed to provide a quantitative, site specific dilution and dispersion model to aid in determination of appropriate dilution factors per guidance provided in the California Ocean Plan (2009).

1.0 REGULATORY BACKGROUND

On October 18, 2004, the State Water Resources Control Board (State Water Board) notified the City of San Diego, as a responsible party, to cease storm water and non point source waste discharges into Areas of Biological Significance (ASBS) or to request an exception from the California Ocean Plan waste discharge prohibition. On December 15, 2004, the City of San Diego requested an exception for ASBS No. 29. On March 20, 2012, the State Water Board adopted Resolution No 2012-0012, approving an exception to the California Ocean Plan, *General Exception to the California Ocean Plan for Areas of Special Biological Significance Waste Discharge Prohibition for Storm Water and Nonpoint Source Discharges, with Special Protections* (herein referred General Protections). These general protections are in accordance with the Porter-Cologne Water Quality Control Act, California Water Code §13000 et seq., and implementing regulations, including the current Ocean Plan (2009).

Under the California Ocean Plan (2009), Section III, C.4.a, Effluent limitations for water quality objectives listed in Table B shall be determined through the use of the following equation:

$$Ce = Co + Dm (Co - Cs)$$

where (in ug/l):

Ce	=	the effluent concentration limit,
Co	=	the concentration (water quality objective) to be met at the completion of initial dilution,
Cs	=	background seawater concentration (references in Table C of Ocean Plan)
Dm	=	minimum probable initial dilution expressed as parts seawater per part wastewater.

Furthermore, the Ocean Plan provides guidance for the identification of dilution models for use in determining the initial dilution, Dm. Specifically, per Section III, C.4.a;

> *The Executive Director of the SWRCB shall identify standard dilution models for use in determining Dm, and shall assist the Regional Board in evaluating Dm for specific waste discharges. Dischargers may propose alternative methods of calculating Dm, and the Regional Board may accept such methods upon verification of its accuracy and applicability.*

2.0 HYDRODYNAMIC MODELING DESIGN

This study is designed to provide site specific dilution and dispersion model results for ASBS No. 29 to the San Diego Regional Water Quality Control Board (SDRWQCB). The effluents from three permitted outfalls within the La Jolla ASBS were studied using the SEDXPORT hydrodynamic modeling system. The model is designed to numerically simulate dry weather and wet weather case scenarios. The dilution study incorporated historical site specific outfall data on water mass boundary properties (bathymetry, salinity, temperature, ocean level/tides) and forcing functions (waves, currents and winds).

The SEDXPORT modeling system was developed at Scripps Institution of Oceanography (SIO) for the US Navy's *Coastal Water Clarity System* and *Littoral Remote Sensing Simulator*. The model has been reviewed and vetted by multiple regulatory agencies and has been calibrated for six previous water quality projects in the Southern California Bight. Due to the large number of oceanographic measurements that have been made in ASBS No 29 and the adjacent ASBS No 31, the model for this site is considered particularly robust. These include long term water quality, wind, wave, water height, tidal and temperature measures at the SIO pier as well as multiple special studies of currents and storm water inputs into La Jolla Bay and environs.

This model was selected as the most relevant model available for the following reasons;

1) The State Water Board required SIO to perform a similar study to determine the initial dilution and dispersion of the discharge during storm and non-storm periods at the adjacent ASBS No. 31 (San Diego-Scripps) as a requirement of SIO's NPDES Permit Exception, Order No. R9-2005-0008. SIO used the SEDXPORT modeling system to fulfill this requirement and the study was provided on February 9 2007 to the San Diego Regional Board and the California SWRCB for approval.

2) The model design and results of the SIO study were evaluated by the Natural Water Quality Committee (NWQC), and deemed appropriate (Summation of Findings, Natural Water Quality Committee, 2006-2009, SCCWRP Technical Report 625, September 2010). The NWQC was a scientific oversight committee established as a requirement of the SIO permit to evaluate monitoring results and special studies. With technical review and input from the NWQC, the SDRWQCB revised the initial 2:1 dilution factor in the initial waste discharge requirements (WDR) to a 7:1 dilution factor (representative of the minimum dilution ratio) and this 7:1 dilution factor was incorporated into the November 2008 revision of SIO's permit.

3) This study utilizes the same modeling program (SEDXPORT) and modeling assumptions and integrating the most recent long term historical trend data. In addition, the model calibration and outputs were generated by the same SIO scientists (Scott Jenkins, Ph.D. and Joseph Wasyl) that published the SDRWQCB approved 2007 SIO study.

4) The mass flow model inputs are based on actual discharge data (flow and mass) measured from permitted outfalls; SDL-157, SDL-062 and SDL-186. To model the beach discharges, flow and chemistry results collected in November 2011 provided data representative of the worse case proxy scenario. These data were merged with the long term probability assessment in order to bracket possible dilution outcomes. The study provide the same data endpoints (dilution ranges of the outer far field [>-10m] of ASBS No. 29 and the near field [<-10m] surf zone dilution ranges) for wet weather discharges under various ocean mixing conditions as used in the aforementioned SIO study. In addition, the extreme worse case probabilities (high storm water flow/low surf and currents) at the zone of initial dilution (ZID) were evaluated to determine minimum dilution factors.

3.0 HYDRODYNAMIC MODELING RESULTS

On behalf of the City of San Diego, AMEC Earth and Infrastructure, Inc. contracted with Dr. Scott A Jenkins Consulting to provide hydrodynamic modeling of storm drain discharges in the vicinity of ASBS No.29. The complete hydrodynamic modeling report is provided as Attachment A.

In order to model possible dilution scenarios, long term trends for boundary and force function data (i.e., 32 year record from 1980-2012) provides a representative basis for the model to incorporate the Pacific Decadal Oscillation (aka alternating periods of strong and weak El Niño). Site specific storm water flow and mass data (three events spanning from November 2011 to March 2012) were obtained from three monitored outfalls within ASBS No 29; SDL-157, SDL-062 and SDL-186.

These data sets (parameters for long term functions and outfall flux/hydrographs) are coupled and the model produces the dilution and dispersion outcomes (aka ranges of dilution factors) by analyzing the possible daily outcomes (maximum and minimum) of the input variables over 32 years (a total of 11,688 distinct combinations). This analysis was performed for each of the outfalls.

These probabilities can be plotted as a density functions and the dilution factors presented relative to the likelihood of occurrence. The resultant density plots span from the highest modeled dilution factor (low flow/high energy ocean conditions) to the lowest “worst case” dilution (high storm water flow/calm ocean conditions). The likelihood of occurrence corresponding to these conditions decreases for both the best and worst case scenarios as the model search criteria plots the possible combinations. See Attachment A, Table 3 for search criteria.

These wet weather worse case scenarios for the dilution and mass are plotted as dilution contour maps, by merging the results for both the offshore and surfzone bathymetry regions. See Attachment A, Figures 30-36. In general, dilutions for storm water range between a minimum of 10^2 to 10^4 in the nearshore and dilution magnitude of 10^4 to 10^7 characterize the outer half offshore of ASBS No. 29.

In the immediate zone of initial dilution, where there is irreversible turbulent mixing, the dilution factor for 90% of the potential outcomes produced a minimum dilution of 20 to 1 for the 48-inch Outfall SDL-157, and 15 to 1 for the 72-inch SDL-062 outfall using the worse case November 2011 storm event. Of the three flow monitored outfalls, the 36-inch SDL-186 (the Devil's slide) has significantly lower storm water discharges, and therefore correspondingly higher near shore dilution factors. Dilution factors for the SDL-186 near shore area were in the range of 10^3.

The largest and single contributing dilution footprint was found at SDL-062, the discharge at the end of Avenita de la Playa Street. This is the largest outfall within ASBS No. 29 and drains the largest fraction of the La Jolla Shores watershed. Therefore, this outfall generates the lowest modeled dilution factors. The median outcome minimum surfzone dilution for SDL-062 is 22 to 1. The minimum worst case value for SDL062, with a probability of occurrence of 0.13%, is 13 to 1 (12.6:1 calculated value) in a peak storm water flow and low energy ocean mixing condition.

4.0 STUDY CONCLUSIONS

Using the SEDXPORT hydrodynamic model, the storm water discharges from monitored outfalls into the La Jolla ASBS No 29 generated dilution factors ranging from 10^2 in the near shore, to 10^7 in the seaward boundary during wet weather. Further resolution of the model at the zone of initial dilution (ZID) produced a worse case dilution factor of 15 to 1 for 90% of the possible outcomes for the largest discharge outfall, SDL-062. The extreme worst case (0.13% probability in conditions of high discharge, calm sea state) generated a 13:1 dilution factor for this outfall.

It was this extreme worse case dilution value that was approved by the SDRWQCB as described in hydrodynamic modeling section, item 2 above.

In addition, it is fortuitous that the lowest flow outfall, SDL-186 at the Devils Slide, has the most high value, hard bottom marine habitat and that dilution outcomes are greater than 3 orders of magnitude in the nearshore.area There is a hard bottom tidal flat in the immediate vicinity of the Devils Slide intertidal zone that is exposed at low tide, that may be subject to acute fresh water exposure during storm events. Conversely, the higher flow SDL-062 discharges across the southern border of La Jolla Shores beach and discharges into the surf zone onto sandy soft bottom marine habitat.

The high dilution factors expected in the vicinity of the Devils Slide area of ASBS No.29 is consistent with other localized water quality measurements during storm water discharge from Outfall SDL-186. These include the La Jolla Shores ASBS Protection Implementation Program administered under Prop 84 funding (Grant agreement No 10-413-550, La Jolla Shores Watershed Management Group [LJSWMG], 2008) and, ASBS No. 29 compliance monitoring under the General Protections, and a corresponding bioaccumulation study (AMEC, 2013).

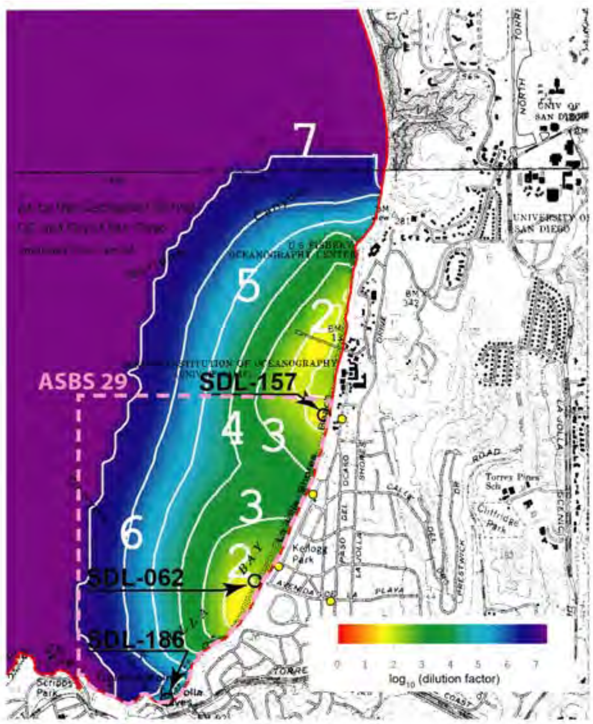
APPENDIX A

HYDRODYNAMIC MODELING OF STORM DRAIN DISCHARGES IN THE NEIGHBORHOOD OF ASBS 29 IN LA JOLLA, CALIFORNIA

This page intentionally left blank

HYDRODYNAMIC MODELING OF STORM DRAIN DISCHARGES IN THE NEIGHBORHOOD OF ASBS 29 IN LA JOLLA, CALIFORNIA

by: Scott A. Jenkins, Ph. D. and Joseph Wasyl



Submitted by:

Dr. Scott A. Jenkins Consulting
14765 Kalapana Street
Poway, CA 92064

Submitted to:
City of San Diego
Transportation and Storm Water Department
9370 Chesapeake Dr., Suite 100
San Diego, CA 92123, USA

Final: 12 May 2013

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS (Cont.)

Page

LIST OF FIGURES (Cont.)

TABLE OF CONTENTS (Cont.)

Page

LIST OF FIGURES (Cont.)

TABLE OF CONTENTS (Cont.)

Page

LIST OF FIGURES (Cont.)

TABLE OF CONTENTS (Cont.)

LIST OF APPENDICES

EXECUTIVE SUMMARY

The La Jolla Shores Coastal Watershed consists of 1,639 acres, primarily residential and institutional (e.g., the University of California, San Diego [UCSD] campus) land uses. Sub-drainages within the watershed boundary drain west into two *areas of special biological significance* (ASBS): the San Diego– Scripps ASBS (ASBS 31) and the La Jolla ASBS (ASBS 29). The majority of the runoff from the watershed is conveyed through a network of storm drains before it is discharged at several locations along the beach. Hydrodynamic model analysis of the dilution and dispersion of shoreline discharges into ASBS 31 was previously done by Jenkins and Wasyl (2007) for the University of California San Diego. The present hydrodynamic analysis applies those same analysis methods in evaluating shoreline discharges of storm water into nearshore waters surrounding ASBS 29. The central and largest portion of the La Jolla watershed affecting ASBS 29 drains to only two storm drain outfalls, SDL-157 at the northern boundary of ASBS 29 and SDL-062 at Avenida de la Playa. During the 2011-2012 wet weather monitoring season (October 1, 2011 through April 30, 2012), storm water samples were collected from outfall discharge and receiving water mixing zone monitoring locations during three events (AMEC, 2012). These data were used to initialize the source loading inputs to a hydrodynamic model (***SEDXPORT***) used in the present receiving water analysis of storm water dilution and dispersion.

The ***SEDXPORT*** hydrodynamic modeling system was developed at Scripps Institution of Oceanography for the US Navy's *Coastal Water Clarity System* and *Littoral Remote Sensing Simulator*. This model has been peer reviewed multiple agencies and has been calibrated and validated in the Southern California Bight for six previous water quality and design projects.

Based on previous protocols established with the San Diego Regional Water Quality Control Board for storm drain dilution analysis in ASBS 31 (Jenkins and Wasyl, 2007), the numerical modeling study of the beach discharges is based on a wet weather worst-case scenario with a long-term probability assessment to bracket the envelope of the possible outcomes. Among the three wet-weather events monitored by AMEC (2012),Wet Weather Event-1 (20-21 November, 2011) comes closest to matching a computer search of marine environmental conditions for the wet-weather worst-case proxy. Analysis of the worst-case wet weather proxy was conducted over the entire coastal domain of La Jolla Bay and the Torrey Pines Littoral Sub-Cell.

The footprint of the dilution field for the wet weather worst case scenario, (involving the simultaneous discharge from storm drains SDL-157, SDL-062 and SDL-186), spreads about a kilometer seaward of the shoreline and about 2 kilometers along the shoreline, covering most of ASBS 29 and intruding into the southern portion of ASBS 31. The dilution factors of storm water in both ASBS 29 and ASBS 31 range between a minimum of 10^2 near shore to 10^7 along the seaward boundaries during the wet weather worst case. Dilution factors of 10^4 to 10^7 characterize the outer one-half of ASBS 29, while dilution factors 10^2 to 10^4 characterize the inner one-half. In the immediate neighborhood of SDL-157 and SDL-062, very near the shoreline of ASBS 29, dilution factors are as low as 40 to one.

The preponderance of the storm water discharge plume during the wet weather worst case scenario is from SDL-157 and SDL-062, while SDL-186 at Devil's Slide makes a very minor contribution. This is a fortuitous outcome in the sense that a significant amount of high-value, hard bottom marine habitat lives in the southern end of ASBS 29, and in the immediate neighborhood of SDL-186. Dilution factors of storm water discharged from SDL-186 are at most 10^3 near the shore in the nearfield of SDL-186, and more typically 10^4 to 10^5 elsewhere along the bluff-faced shoreline of the southern end of ASBS 29. The largest single contributing footprint to the combined storm water plume appears to be from SDL-062 at the end of Avendia De La Playa that produces a large patch with 10^2 minimum dilution very near the shoreline that spreads south along the shore to the La Jolla Beach and Tennis Club. The region influenced by this feature from SDL-062 is sandy, soft-bottom marine habitat. SDL-157 produces another patch with 10^2 minimum dilution, but this feature moves offshore and intrudes into the southern end of ASBS 31, where again the marine habitat is a sandy soft-bottom type.

Maximum concentrations of storm water born total suspended solids (TSS) in the receiving water due to SDL-186 storm water discharges are at most 0.001 mg/L near the shore in the nearfield of SDL-186, and more typically 0.0001 mg/L elsewhere along the bluff-faced shoreline of the southern end of ASBS 29, where significant high-value, hard bottom marine habitat resides. The highest TSS concentrations in the receiving water around SDL-062 (at the end of Avendia De La Playa), were found to be 3 mg/L in a nearshore patch that that spreads south along the shore to the La Jolla Beach and Tennis Club. Most of the shoreline impacted by SDL-157 experiences maximum TSS concentrations from storm water in the range of 0.3 mg/L to 0.03 mg/L, all of which is sandy soft- bottom marine habitat. At the northern end of ASBS 29, and extending into ASBS 31, maximum TSS concentrations in the receiving from SDL-157 are in the range of 4 mg/L to 0.4 mg/L, the highest anywhere in La Jolla Bay. These relatively higher TSS values are probably a consequence of the steep land forms that comprise the watershed of SDL-157, with portions of both developed and undeveloped coastal bluffs. Regardless, the marine habitat subjected to these TSS loadings from SDL-157 is a sandy, soft-bottom type.

Because of the disparity in length scales between dilution in the offshore region versus the surfzone, a separate analysis was performed on a nested fine scale grid covering the surf zone for a long shore reach that extended 1000 ft (305 meters) either side of the beach outfalls. The size of this grid was based on precedent already set for the definition of a zone of initial dilution (ZID) by the Regional Water Quality Control Board, San Diego. Thirty two years of receiving water variables (involving 11,688 distinct combinations) were input for daily simulations of dilution. The ZID was searched for the minimum dilution after averaging across the width of the surf zone. Source loading for these simulations were based on the discharges measured during Wet-Weather Event-1 from outfalls SDL-157, SDL-062 and SDL-186. The surfzone dilution results are then compared to minimum dilution on the sea surface at an offshore control point in ASBS 29 where local water depth is -10 m MSL.

This probability analysis procedure (based on the historic combination of 11,688 environmental variables) inevitably produces some combinations of wet weather discharges with dry weather ocean mixing conditions, and thereby results in certain worst case dilution scenarios that are more extreme, with even less dilution, than the wet weather worst case proxy derived from the monitoring program (Wet-Weather Event-1). For storm drain SDL-157, the median outcome for minimum dilution factor within the surfzone ZID is 32 to 1; however the potential range of minimum dilution goes as high as 88 to 1, and as low as 15 to 1. Low energy, dry weather ocean mixing produced the 15 to 1 minimum surfzone dilution outcome, which had a probability of occurrence of 0.13%. Altogether, 90% of the potential outcomes produce minimum dilutions in the surfzone ZID greater than 20 to 1.

Minimum surfzone dilutions in the ZID off SDL-062 are slightly less. This is due to the wave shadow found chronically in the refraction pattern of La Jolla Shores in the neighborhood of Avendia De La Playa. The minimum surfzone dilution of discharges from SDL-062 is found to be 12.6 to 1, again a low energy, dry weather ocean mixing result. The median outcome of minimum surfzone dilution for SDL-062 is 22 to 1; while the potential range of minimum dilution goes as high as 64 to 1, based on the 32-year historical sequence of receiving water variables. Ninety percent of the potential outcomes for SDL-062 produce minimum surfzone dilutions greater than 15 to 1.

For comparison, probability statistics for minimum dilution at an offshore control point in ASBS 29 (where local water depth is -10 m MSL) discovered a worst case minimum dilution of storm water of 13,200 to 1; while the median outcome is 52,000 to 1. The highest minimum dilutions of storm water at the offshore control point were found to be 4.2×10^5. Consequently, storm water dilution is sufficiently high in the offshore regions of ASBS 29 that concentrations of storm water runoff constituents should be well below quantifiable detection limits.

Hydrodynamic Modeling of Storm Drain Discharges in the Neighborhood of ASBS 29 in La Jolla CA

By: Scott A. Jenkins, Ph. D. and Joseph Wasyl

1.0 INTRODUCTION

The La Jolla Shores Coastal Watershed consists of 1,639 acres, primarily residential and institutional (e.g., the University of California, San Diego [UCSD] campus) land uses. Sub-drainages within the watershed boundary drain west into two *areas of special biological significance* (ASBS): the San Diego– Scripps ASBS (ASBS 31) and the La Jolla ASBS (ASBS 29). The majority of the runoff from the watershed is conveyed through a network of storm drains before it is discharged at several locations along the beach. Hydrodynamic model analysis of the dilution and dispersion of shoreline discharges into ASBS 31 was previously done by Jenkins and Wasyl (2007) for the University of California San Diego. The present hydrodynamic analysis applies those same analysis methods in evaluating shoreline discharges of storm water into nearshore waters surrounding ASBS 29.

The central and largest portion of the La Jolla watershed affecting ASBS 29 drains to a single storm drain outfall that discharges at Avenida de la Playa.

Monitoring within ASBS 29 and the drainage area discharging to the ASBS has been conducted over a number of years at multiple locations. The La Jolla Shores Watershed Urban Runoff Characterization and Watershed Characterization Study (Weston Solutions, Inc., Weston, 2007) was conducted under Proposition 50 - Funding for Public Water Systems (Prop 50). The Weston (2007) study assessed data collected between 2005 and 2007 at locations within the watershed and receiving water to identify constituents of interest (COI) for ASBS 29 for future studies. Monitoring based on the Special Protections Document has been conducted since the 2008-2009 wet weather monitoring season. Monitoring activities were conducted by Weston for the wet weather monitoring seasons from fall 2008 through spring 2011 (Weston, 2009, 2010, 2011). Monitoring by Weston included a variety of sample collection programs, including pre-storm, during-storm, and post-storm sample collections. Sample collection occurred at receiving water mixing zone, outfall discharge, and reference beach locations. A summary of previous studies was published by AMEC Earth and Infrastructure, Inc. (AMEC) in a draft conceptual ASBS 29 Characterization Summary and is provided in Appendix A of AMEC (2012). During the 2011-2012 wet weather monitoring season (October 1, 2011 through April 30, 2012), storm water samples were collected from outfall discharge and receiving water mixing zone monitoring locations during three events (AMEC, 2012). Sediment samples were collected from receiving water mixing zone monitoring locations after the third monitored event. These water quality data were used to initialize the source loading inputs to the hydrodynamic model used in the present storm water dilution and dispersion analysis.

Dilution analysis of three storm drain discharges (SDL-157, SDL-062 and SDL-186) were studied in numerical simulation for wet weather extreme and average case scenarios using the ***SEDXPORT*** hydrodynamic modeling system that was developed at Scripps Institution of Oceanography for the US Navy's *Coastal Water Clarity System* and *Littoral Remote Sensing Simulator*. This model has been peer reviewed multiple times and has been calibrated and validated in the Southern California Bight for 4 previous water quality and design projects (cf. Section 2).

The locations of the monitoring locations storm drain outfalls (SDL-157, SDL-062 and SDL-186) are shown in Figure 1 and are characterized by the following locations and descriptions:

Table 1.
Outfall Discharge Monitoring Locations and Descriptions

Site ID	Pipe Diameter (in)	Number of Events / Sample Type	Latitude	Longitude	Description
SDL-062-OD	72	3 / Flow-weighted Composite	32.85384	-117.25491	Upstream of outfall at intersection of Avenida de la Playa and Paseo del Ocaso.
SDL-157-OD	48	3 / Flow-weighted Composite	32.86278	-117.25429	Upstream of outfall on east side of El Paseo Grande.
SDL-186-OD	36	2 / Time-weighted composite of 2 - 4 grab samples	32.84813	-117.26507	Upstream of outfall on Torrey Pines Road
SDL-063-OD	36	Not monitored	32.85563	-117.25818	End of Vallecitos Road
SDL-063B-OD	36	Not monitored	32.85913	-117.25608	On seawall approximately 300 feet north of northern end of Kellogg Park parking lot

Storm Drain SDL-062 – This 72-inch storm drain discharges approximately 44 % of the La Jolla Shores Coastal Watershed runoff. This percentage was revised from 49 % stated in the Work Plan based on a recalculated drainage area for the modified flow model. The SDL-062 outfall is located at the end of Avenida de la Playa and discharges to La Jolla Shores Beach. Flow-weighted composite samples were collected approximately one-quarter mile upstream of the outfall at a point where more accurate flow data could be collected. The monitoring location was within a manhole at the intersection of Avenida de la Playa and Paseo del Ocaso (Figure 1).

Storm Drain SDL-157 – This 48-inch outfall discharges directly to La Jolla Shores Beach and drains approximately 12 % of the La Jolla Shores Watershed. Flow-weighted samples were collected approximately 180 feet upstream of the outfall at a point where more accurate flow data could be collected. The monitoring location was within a manhole on the eastern side of El Paseo Grande (Figure 1).

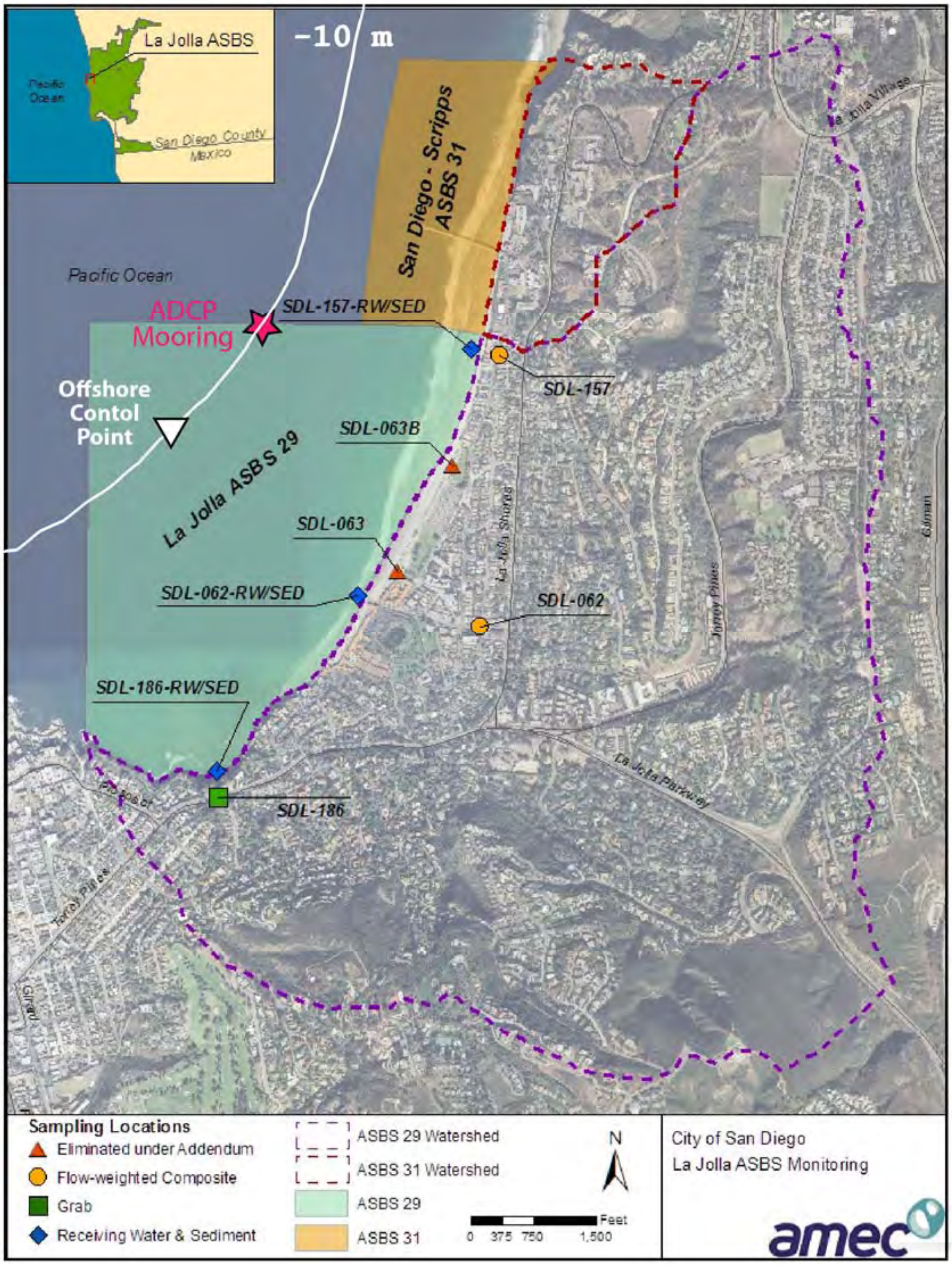


Figure 1. Location Map for La Jolla Storm Drains SDL-157, SDL-062 and SDL-186 in relation to ASBS 29 and ASBS 31, (from AMEC 2012).

Storm Drain SDL-186 – This 36-inch outfall discharges directly to the rocky intertidal area of the ASBS south of La Jolla Shores Beach, known as Devil's Slide. This outfall drains approximately 3 % of the La Jolla Shores Coastal Watershed (Figure 1). This percentage was revised from 4 % stated in the Work Plan (MACTEC. 2011a. & b.)based on a recalculated drainage area for the modified flow model. Access to this outfall is restricted during high tide conditions. Given the lack of consistent, safe access to the outfall, the storm drain upstream was sampled instead. This manhole is located on Torrey Pines Road between Prospect Place and Amalfi Street, approximately 160 feet upstream of the outfall.

The quantification of dilution requires solving the hydrodynamic transport equations for the spatial and temporal variation of dilution factor throughout the effected receiving water. The dilution factor analysis will be evaluated at two distinct worse case scenarios: 1) base flows during dry weather with low mixing rates in the receiving waters due to quiescent ocean/atmosphere conditions; and 2) storm water runoff and discharge during high energy conditions typical of a winter storm event.

It is sensible to bifurcate the model problem into these dry and wet weather scenarios based on the multi-decadal dry/ wet cycles that the regional climate undergoes. The California coast is subject to climate cycles of about 20-30 years duration known as the Pacific/ North American pattern (for atmospheric pressure) or the Pacific Decadal Oscillation (for sea surface temperature). These dry/ wet cycles are apparent in the historic rainfall record of San Diego shown in Figure 2a. A dry period extended from about 1945-1977, followed by an episodically wet period from 1978-1998 that included the occurrence of 6 strong El Niño events (Inman and Jenkins 1999; and Goddard and Graham 1997). Based on the historic duration of these cycles, 1998 was likely the end of the wet cycle of climate in California with a return to the dry climate that prevailed from 1945-1977.

To illustrate the historical evidence for these dry and wet climate cycles, the rainfall record in Figure 2a was analyzed for climate trends using the Hurst (1951, 1957) procedure that was first used for determining decadal climate effects on the storage capacity of reservoirs (Inman and Jenkins, 1999). Climate trends become apparent when the data are expressed in terms of cumulative residuals of rainfall RF_n taken as the continued cumulative sum of departures of annual values of a time series RF_i from their long term mean value RF_a such that $RF_n = \sum_0^n \left(RF_i - RF_a \right)$ where n is the sequential value of the time series. When this procedure was applied to the rainfall record in Figure 2b, dry periods are revealed by segments of the cumulative residuals having negative (downward) slopes while the wet periods have positive (upward) slopes. A dry period is found from 1945-1997, (negative slopes) while a wet period (positive slope) is shown from 1978-1998. The wet period of the climate cycle is more irregular caused by 6 strong El Niño events (water years 1978, 80, 83, 93, 95, and 98) and one 4 year period (1987-1990) of low rainfall. The analysis shows that the average annual rainfall increased by about 38% from the dry to the wet portions of the cycle. Furthermore, both the minimum and

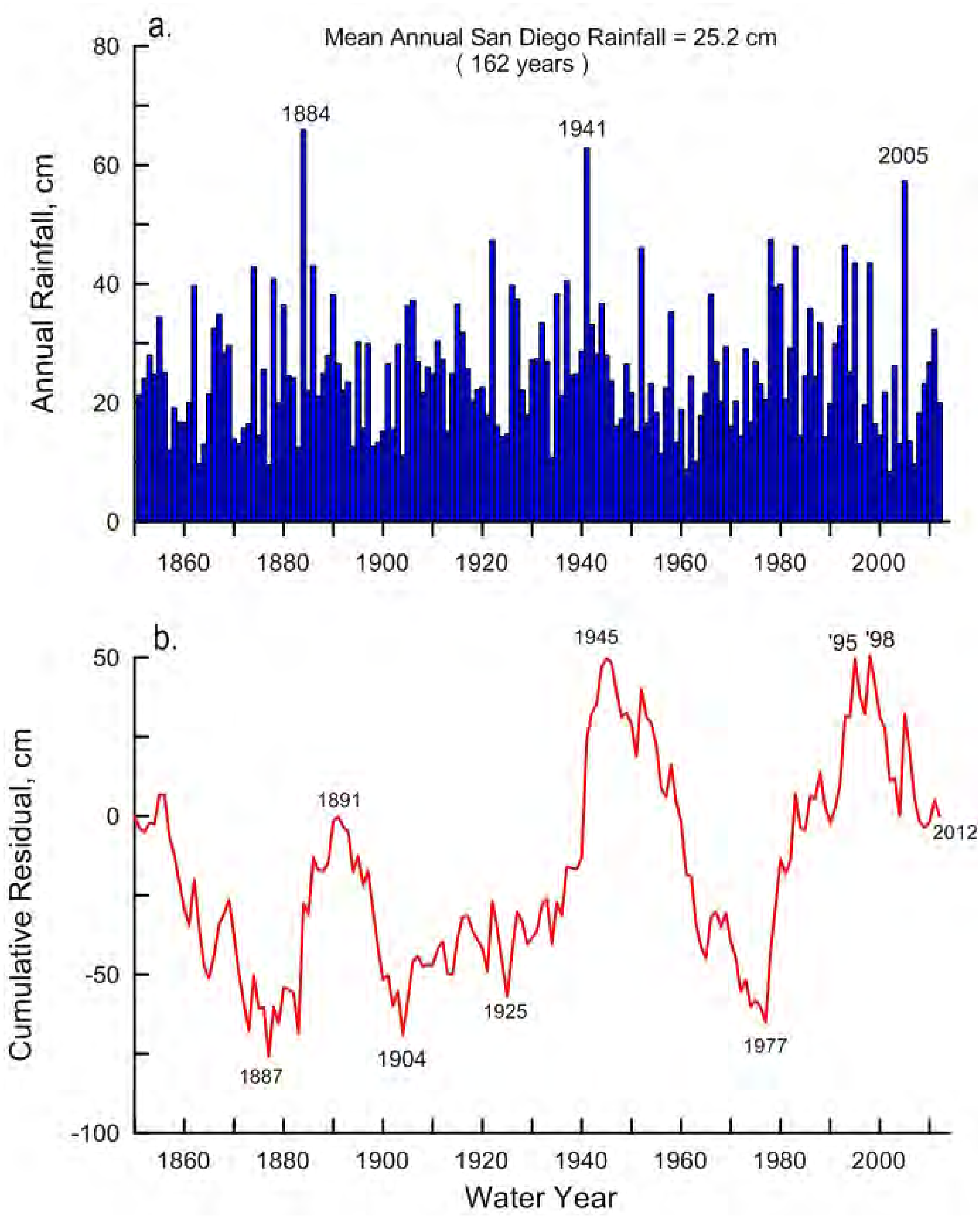


Figure 2. a) Period of record of San Diego rainfall and b) cumulative residual

maximum ranges in rainfall are higher in the wet period, while the averages of the 6 major rainfall events in 21 year periods before and after the climate change (1977/78) are about 8 to 9 inches greater during the wet period. Since 1998, Figure 2b indicates that San Diego climate appears to have regressed into a developing multi-decadal dry period. These sharp distinctions in the statistics of dry vs wet climate periods lead to posing the model problem of discharge dilution in terms of dry and wet extreme case scenarios in order to bracket the envelope of potential variability.

Altogether there are seven primary variables that enter into a solution for resolving the dispersion and dilution of shoreline discharges such as those found along Scripps and La Jolla Shores Beach and the rocky shores of the Devil's Slide area. The statistics of these seven variables all change between dry and wet climate periods. These seven variables may be organized into ***boundary conditions*** **and** ***forcing functions***. The boundary conditions include: ocean salinity, ocean temperature, ocean water levels and discharge flow rates. There is an additional boundary condition associated with offshore bathymetry beyond closure depth (typically greater than 12-15 m) that we will treat as constant. These boundary condition variables are developed in Section 3.1. The forcing function variables include: waves, currents, and winds and are developed in Section 3.2.

Overlapping 32 year long records containing 11,688 consecutive days between 1980 and 2012 are reconstructed in Sections 3.1 and 3.2 for each of the seven controlling variables. We search this 20.5 year period for the historical combination of these variables that give an historic extreme day in the sense of benign ocean conditions that minimize mixing and dilution rates, and a high energy day giving mixing and dilution rates typical of winter storm conditions with rainfall. We then overlay the peak discharge rates of seawater and storm water on those environmental conditions, respectively. The criteria for an extreme dry weather day was based on the simultaneous occurrence of the environmental variables having the highest combination of absolute salinity and temperature during the periods of lowest mixing and advection in the local ocean environment. These conditions coincide with the simultaneous occurrence of the lowest 5% wave, wind, currents, and ocean water levels averaged over a 24 hour period. The extreme wet weather scenarios were found by a statistical search of these records for the simultaneous occurrence of the highest 5% wave, wind, currents, and ocean water levels averaged over a 24 hour period. This procedure produced the model scenarios defined in Section 3.3. We also set up the model in Section 3.3 to solve for the minimum dilution in the surf zone for all 7523 combinations of the 7 controlling variables in order to establish the statistical properties the ***zone of initial dilution*** **(*ZID*)**.

The technical approach used to evaluate these scenarios for historical extremes and average case conditions involved the use of the SEDXPORT hydrodynamic transport models. The pedigree and physics of this modeling system is described briefly in Section 2, with additional details provided in Appendices A &B. The dilution fields simulated by this model are presented in Section 4, giving results for the wer weather extreme in Section 4.1, and the long term minimum dilution in the ZID in Section 4.2. Dilution fields for an offshore control point in ASBS 29 are also given in Section 4.2.

2.0 MODEL DESCRIPTION AND CAPABILITIES

This study utilizes a coupled set of numerical tidal and wave transport models to evaluate dilution and dispersion of the discharges from the three storm drain outfalls discharging into ASBS 29 (Figure 1). The numerical model used to simulate tidal currents in the nearshore and shelf region offshore of La Jolla Shores is the finite element model **TIDE_FEM**. Wave-driven currents are computed from the shoaling wave field by a separate model, **OCEANRDS**. The dispersion and transport of concentrated seawater and storm water discharge by the wave and tidal currents is calculated by the finite element model known as **SEDXPORT**.

The finite element research model, **TIDE_FEM**, (Jenkins and Wasyl, 1990; Inman and Jenkins, 1996) was employed to evaluate the tidal currents within La Jolla Bay and in particular, the ventilation of the ASBS 29. **TIDE_FEM** was built from some well-studied and proven computational methods and numerical architecture that have done well in predicting shallow water tidal propagation in Massachusetts Bay (Connor and Wang, 1974) and along the coast of Rhode Island, (Wang, 1975), and have been reviewed in basic text books (Weiyan, 1992) and symposia on the subject, e.g., Gallagher (1981). The governing equations and a copy of the core portion of the **TIDE_FEM** FORTRAN code are found in Appendix A. **TIDE_FEM** employs a variant of the vertically integrated equations for shallow water tidal propagation after Connor and Wang (1975). These are based upon the Boussinesq approximations with Chezy friction and Manning's roughness. The finite element discretization is based upon the commonly used **Galerkin weighted residual method** to specify integral functionals that are minimized in each finite element domain using a variational scheme, see Gallagher (1981). Time integration is based upon the simple **trapezoidal rule** (Gallagher, 1981).

The computational architecture of **TIDE_FEM** is adapted from Wang (1975), whereby a transformation from a **global** coordinate system to a **natural** coordinate system based on the unit triangle is used to reduce the weighted residuals to a set of order-one ordinary differential equations with constant coefficients. These coefficients (**influence coefficients**) are posed in terms of a **shape function** derived from the natural coordinates of each nodal point in the computational grid. The resulting systems of equations are assembled and coded as banded matrices and subsequently solved by **Cholesky's method**, see Oden and Oliveira (1973) and Boas (1966). The hydrodynamic forcing used by **TIDE_FEM** is based upon inputs of the tidal constituents derived from Fourier decomposition of tide gage records. Tidal constituents are input into the module **TID_DAYS**, which resides in the hydrodynamic forcing function cluster (see Appendix B for a listing of **TID_DAYS** code). **TID_DAYS** computes the distribution of sea surface elevation variations in La Jolla Bay based on the tidal constituents derived from the Scripps Pier tide gage station (NOAA #941-0230). Forcing for **TIDE_FEM** is applied by the distribution in sea surface elevation across the deep water boundary of the computational domain.

Wave driven currents were calculated from wave measurements by Scripps SAS station at Torrey Pines (Pawka, 1982) and by the CDIP arrays and/or buoys at Scripps Pier, La Jolla Bay, Huntington Beach, San Clemente, and Oceanside, CA, see CDIP (2005). These measurements were back refracted out to deep water to correct for island sheltering effects between the monitoring sites and Scripps Beach. The waves were then forward refracted onshore to give the variation in wave heights, wave lengths and directions throughout the nearshore around Scripps Beach. The numerical refraction-diffraction code used for both the back refraction from these wave monitoring sites out to deep water, and the forward refraction to the La Jolla Shores site is **OCEANRDS** and may be found in Appendix C. This code calculates the simultaneous refraction and diffraction patterns of the swell and wind wave components propagating over bathymetry replicated by the **OCEANBAT** code found in Appendix D. **OCEANBAT** generates the associated depth fields for the computational grid networks of both **TID_FEM** and **OCEANRDS** using packed bathymetry data files derived from the National Ocean Survey (NOS) depth soundings. The structured depth files written by **OCEANBAT** are then throughput to the module **OCEANRDS**, which performs a refraction-diffraction analysis from deep water wave statistics. **OCEANRDS** computes local wave heights, wave numbers, and directions for the swell component of a two-component, rectangular spectrum.

The wave data are throughput to a wave current algorithm in **SEDXPORT** (Appendix E) which calculates the wave-driven longshore currents, v(*r*). These currents were linearly superimposed on the tidal current. The wave-driven longshore velocity, v(*r*), is determined from the longshore current theories of Longuet-Higgins (1970). Once the tidal and wave driven currents are resolved by **TIDE_FEM** and **OCEANRDS**, the dilution and dispersion of storm water runoff and seawater discharge is computed by the stratified transport algorithms in **SEDXPORT**. The **SEDXPORT** code is a time stepped finite element model which solves the advection-diffusion equations over a fully configurable 3-dimensional grid. The vertical dimension is treated as a two-layer ocean, with a surface mixed layer and a bottom layer separated by a pycnocline interface. The code accepts any arbitrary density and velocity contrast between the mixed layer and bottom layer that satisfies the Richardson number stability criteria and composite Froude number condition of hydraulic state.

The combined discharge of seawater and storm water from the 3 La Jolla Shores beach outfalls is represented as sources in the surface mixed layer. The source initializations for these beach discharges are handled by a companion dilution code called **MULTINODE** (Appendix F) that couples the computational nodes of **TIDE_FEM** and **OCEANRDS** with **SEDXPORT**. The codes do not time split advection and diffusion calculations, and will compute additional advective field effects arising from spatial gradients in eddy diffusivity, (the so-called "gradient eddy diffusivity velocities" after Armi, 1979). Eddy mass diffusivities are calculated from momentum diffusivities by means of a series of Peclet number corrections based upon TSS and TDS mass and upon the mixing source. Peclet number corrections for the surface and bottom boundary layers are derived from the work of Stommel (1949) with modifications after Nielsen (1979), Jensen and Carlson (1976), and Jenkins and Wasyl (1990). Peclet number correction for the wind-induced mixed layer diffusivities are calculated from algorithms developed by Martin and Meiburg (1994), while Peclet number corrections to the interfacial shear at the pycnocline are derived from Lazara and Lasheras (1992a;1992b). The momentum diffusivities to which these Peclet number corrections are applied are due to Thorade (1914), Schmidt (1917), Durst (1924), and Newman (1952) for the wind-induced mixed layer turbulence and to Stommel (1949) and List, et al. (1990) for the current-induced turbulence.

In its most recent version, **SEDXPORT** has been integrated into the Navy's Coastal Water Clarity Model and the Littoral Remote Sensing Simulator (LRSS) (see Hammond, et al., 1995). The **SEDXPORT** code has been validated in mid-to-inner shelf waters (see Hammond, et al., 1995; Schoonmaker, et al., 1994). Validation of the **SEDXPORT** code was shown by three independent methods: 1) direct measurement of suspended particle transport and particle size distributions by means of a laser particle sizer; 2) measurements of water column optical properties; and, 3) comparison of computed stratified plume dispersion patterns with LANDSAT imagery.

Besides being validated in coastal waters of Southern California, the **SEDXPORT** modeling system has been extensively peer reviewed. Although some of the early peer review was confidential and occurred inside the Office of Naval Research and the Naval Research Laboratory, the following is a listing of 6 independent peer review episodes of **SEDXPORT** that were conducted by 8 independent experts and can be found in the public records of the State Water Resources Control Board, the California Coastal Commission and the City of Huntington Beach.

1997 – Reviewing Agency: State Water Resources Control Board
Project: NPDES 316 a/b Permit renewal, Scripps Beach, Carlsbad, CA
Reviewer: Dr. Andrew Lissner, SAIC, La Jolla, CA

1998 – Reviewing Agency: California Coastal Commission
Project: Coastal Development Permit, San Dieguito Lagoon Restoration
Reviewers: Prof. Ashish Mehta, University of Florida, Gainesville; Prof. Paul Komar, Oregon State University, Corvallis; Prof. Peter Goodwin, University of Idaho, Moscow

2000 – Reviewing Agency: California Coastal Commission
Project: Coastal Development Permit, Crystal Cove Development
Reviewers: Prof. Robert Wiegel, University of California, Berkeley; Dr. Ron Noble, Noble Engineers, Irvine, CA

2002 – Reviewing Agency: California Coastal Commission
Project: Coastal Development Permit, Dana Point Headland Reserve
Reviewers: Prof. Robert Wiegel, University of California, Berkeley; Dr. Richard Seymour, University of California, San Diego

2003 – Reviewing Agency: City of Huntington Beach
Project: EIR Certification, Poseidon Desalination Project
Reviewer: Prof. Stanley Grant, University of California, Irvine

2006 – Reviewing Agency: Regional Water Quality Control Board, San Diego Region
Project: UCSD Storm Drain Dilution Study and ASBS 31 Impacts
Reviewer: Mr.John Robertus and Dr. Charles Chen, RWQCB

SEDXPORT has been built in a modular computational architecture with a set of subroutines divided into two major clusters: 1) those which prescribe hydrodynamic forcing functions; and, 2) those which prescribe the mass sources acted upon by the hydrodynamic forcing to produce dispersion and transport. The cluster of modules for hydrodynamic forcing ultimately prescribes the velocities and diffusivities induced by wind, waves, and tidal flow for each depth increment at each node in the grid network. The subroutines **RIVXPORT** and **BOTXPORT** in **SEDXPORT** solve for the mixing and advection of the seawater and buoyant storm water discharge in response to the wave and tidal flow using an rms vorticity-based time splitting scheme. Both **BOTXPORT** and **RIVXPORT** solve the eddy gradient form of the advection diffusion equation for the water column density field:

$$\frac{\partial \rho}{\partial t} = \left(\vec{u} \bullet \nabla \varepsilon\right) \bullet \nabla \rho - \varepsilon \nabla^2 \rho + \rho_0 Q_0 \tag{1}$$

where $\vec{u}$ is the vector velocity from a linear combination of the wave and tidal currents, ε is the mass diffusivity, Λ is the vector gradient operator and ρ is the water mass density in the nearshore dilution field; and ρ_0 is the density of the water discharged by the outfall at a flow rate $\frac{dV_0}{dt}$. The density of the discharge is a function of the bulk density of the suspended solids ρ_s and the density of the discharge fluid ρ_f that transports those solids, or:

$$\rho_0 = \rho_s + (1 - N)\,\rho_f = \rho_q N + (1 - N)\,\rho_f \tag{2}$$

where N is the volume concentration of suspended solids equal to the ratio of suspended solids to sample volume; and ρ_q = 2.65 g/cm^3 is the density of the suspended solid particles taken to be fine-grained quartz.

Both the density of the receiving water ρ and the density of the discharge fluid ρ_f is a function of temperature, *T*, and salinity, *S*, according to the equation of state expressed in terms of the specific volume, $\alpha = 1/\rho$ and $\alpha_f = 1/\rho_f$ or:

$$\begin{aligned} \frac{d\alpha}{\alpha} &= \frac{1}{\alpha}\frac{\partial\alpha}{\partial T}dT + \frac{1}{\alpha}\frac{\partial\alpha}{\partial S}dS \\ \frac{d\alpha_f}{\alpha_f} &= \frac{1}{\alpha_f}\frac{\partial\alpha_f}{\partial T}dT + \frac{1}{\alpha_f}\frac{\partial\alpha_f}{\partial S}dS \end{aligned} \tag{3}$$

The factor $1/\alpha\ \partial\alpha/\partial T$, which multiplies the differential temperature changes, is known as the coefficient of thermal expansion and is typically 2 x 10^{-4} per ^{o}C for seawater; the factor $1/\alpha\ \partial\alpha/\partial S$ multiplying the differential salinity changes, is the coefficient of saline contraction and is typically 8 x 10^{-4} per part per thousand (ppt) where 1.0 ppt = 1.0 g/L of total dissolved solids (TDS). For a standard seawater, the specific volume has a value $\alpha = 1/\rho$ = 0.97264 cm^3/g. If the percent change in specific volume by equation (3) is less than zero, then the water mass is heavier than standard seawater, and lighter if the percent change is greater than zero.

The dilution ratio is given by the volume concentration of the discharged suspended solids in the receiving water and follows from the sediment continuity equation:

$$\frac{\partial N}{\partial t} = (\vec{u} \bullet \nabla\varepsilon) \bullet \nabla N - \varepsilon\nabla^2 N - W_0 \frac{dN}{dz} \tag{4}$$

where W_0 is the settling velocity of suspended particles. It is necessary to correct dilution ratio calculations for the loss of suspended particles due to deposition, so that loss is not included as a pseudo-dilution. The deposition flux is the net between settling and re-suspension and is found from a sub-set of solutions to (4) at the seabed as originally posed for steady flow by Krone (1962) and expanded to oscillatory flow by Jenkins and Wasyl (1990):

$$\Lambda = \frac{-K_s g N_c \left[W_0 N_c - \varepsilon \left(\frac{\partial N}{\partial z} \right)_{z=0} \right]}{(1 - N_c / N_s)} \tag{5}$$

where K_s = 4 x 10^{-14} sec is the sedimentation coefficient after Fujita (1962); *g* is the acceleration of gravity; N_c is the volume concentration at the top of the wave boundary layer; and N_s is the volume concentration of the seabed sediments. If N_0 is the volume concentration of suspended solids at the point of discharge (end-of-pipe); and $N(x, y, z)$ is the volume concentration at any location in the receiving waters, then the dilution factor at that location (corrected for deposition losses) is:

$$D(x,y,z) = \frac{N_0}{N(x,y,z)} - \frac{\Lambda}{W_0} \quad (6)$$

Hence, net deposition ($\Lambda > 0$) in the ASBS acts to increase the retention of discharged particulate, and consequently reduces the apparent dilution factor.

In (1) and (4) the term $\nabla\varepsilon$ acts much like an additional advective field in the direction of high to low eddy diffusivity. This additional "gradient eddy diffusivity velocity" is the result of local variations in current shear and wave boundary layer thickness. Both are bathymetrically controlled and the latter is associated with the refraction/diffraction pattern and is strongest in the wave shoaling region nearshore.

The settling velocity W_0 in (4) and (5) is particle size dependent. The **SEDXPORT** code is configured to accept up to nine particle size bins which for fine-grained particulate are assigned according to the particle size distribution after Jerlov (1976):

$$N(d) = \hat{N}\left(\frac{d}{\hat{d}}\right)^{\gamma} \quad (7)$$

where

$$N_0 = \sum_d N(d) \quad (8)$$

Here $\hat{d}$ is the reference grain size, typically taken as 1.0 microns; and γ is the slope of the particle size distribution on log-log scale where $\gamma \cong 2.5$ for the global average, and $\hat{N}$ is the volume concentration of the reference grain size that is adjusted such that (8) satisfies the measure value N_0 at end-of-pipe.

Solutions for the density and concentration fields calculated by the **SEDXPORT** codes from equations (1)-(4), (7) & (8) are through put to the dilution codes of **MULTINODE** to resolve dilution factors according to (5) & (6). These codes solve for the dilution factor (mixing ratio) for each cell in the finite element mesh of the nearshore computational domain based on a mass balance between imported exported and resident mass of that cell (see Appendix F). The diffusivity, ε, in (1) controls the strength of mixing and dilution of the seawater and storm water constituents in each cell and varies with position in the water column relative to the pycnocline interface. Vertical mixing includes two mixing mechanisms at depths above and below the pycnocline: 1) fossil turbulence from the bottom boundary layer, and 2) wind mixing in the surface mixed layer. The pycnocline depth is treated as a zone of hindered mixing and varies in response to the wind speed and duration. Below the pycnocline, only turbulence from the bottom wave/current boundary layer contributes to the local diffusivity. Nearshore, breaking wave activity also contributes to mixing. The surf zone (zone of initial dilution) is treated as a line source of turbulent kinetic energy by the subroutine **SURXPORT** (Appendix E). This subroutine calculates seaward mixing from fossil surf zone turbulence, and seaward advection from rip currents embedded in the line source. Both the eddy diffusivity of the line source and the strength and position of the embedded rip currents are computed from the shoaling wave parameters evaluated at the breakpoint, as throughput of **OCEANRDS**.

3.0 MODEL INITIALIZATION

Uninterrupted, long-term monitoring of ocean properties has been conducted at the nearby Scripps Pier. The Scripps Pier has been the site of both a NOAA tide gage station (NOAA #941-0230) as well as a monitoring station of the Coastal Data Information Program. It has also been the site where many new monitoring techniques have been developed and validated. We will take advantage of these long term observations to develop the data bases for initializing the boundary conditions and forcing functions used in the model. Statistical searches of these data bases will be performed to extract the dry and wet weather extreme case scenarios.

These dry and wet weather model scenarios are proxies for the extremes of the long term climate variability of the region, and serve to bracket the envelope of potential discharge effects on ASBS 29. Climate variability begins with seasonal variations in Earth's exposure to the sun, producing inter-annual variations in atmospheric pressure fields which in turn cause the Earth's inter-annual seasons. Upon occasion, the typical seasonal weather cycles are abruptly and severely modified on a global scale. These intense global modifications are signaled by anomalies in the pressure fields between the tropical eastern Pacific Ocean and Australia/Malaysia known as the ***El Niño Southern Oscillation***, commonly referred to as ***ENSO***. The intensity of the oscillation is often measured in terms of the ***Southern Oscillation Index (SOI)***, defined as the monthly mean sea level pressure anomaly in mb normalized by the standard deviation of the monthly mean pressures for the period 1951-1980 at Tahiti minus that at Darwin, Australia. (Because the SOI is a ratio of terms that all have units of atmospheric pressure, it is a non-dimensional number).

The Southern Oscillation is in turn, modulated over multi-decadal periods by the ***Pacific Decadal Oscillation***, which results in alternating decades of strong and weak El Niño. The long-term variability of the Pacific Decadal Oscillation (PDO) is shown in Figure 3 and the cumulative residual of the Southern Oscillation Index, between 1882 and 1996, is plotted in Figure 4. Southern Oscillation effects give rise to enhancements and protractions of the inter-annual seasonal cycles, and their two extremes are referred to as El Niño (SOI negative) and La Niña (SOI positive). Inspection of Figures 3 and 4 reveals a number of large positive oscillations in the SOI between 1944 and 1978 corresponding to La Niña dominated climate; and a series of very large negative oscillations occurring between 1978 and 1998 which correspond with El Niño dominated climate. Along the southern California coast, a period of mild-stable weather occurred during the 30 years between the mid-1940's and mid-1970's when La Niña dominated pressure systems prevailed. The average SOI for this period was +0.1, with strong La Niña events in 1950, (SOI = +1.4); 1955/56, (+1.2); 1970/71, (+1.0); 1973/74, (+1.0); and 1975/76 (+1.4). Winters were moderate with low rainfall (see Figure 2), and winds were predominantly from the west-northwest. The principal wave energy was from Aleutian lows having storm tracks which usually did not reach southern California. Summers were mild and dry with the largest summer swells coming from very distant southern hemisphere storms.

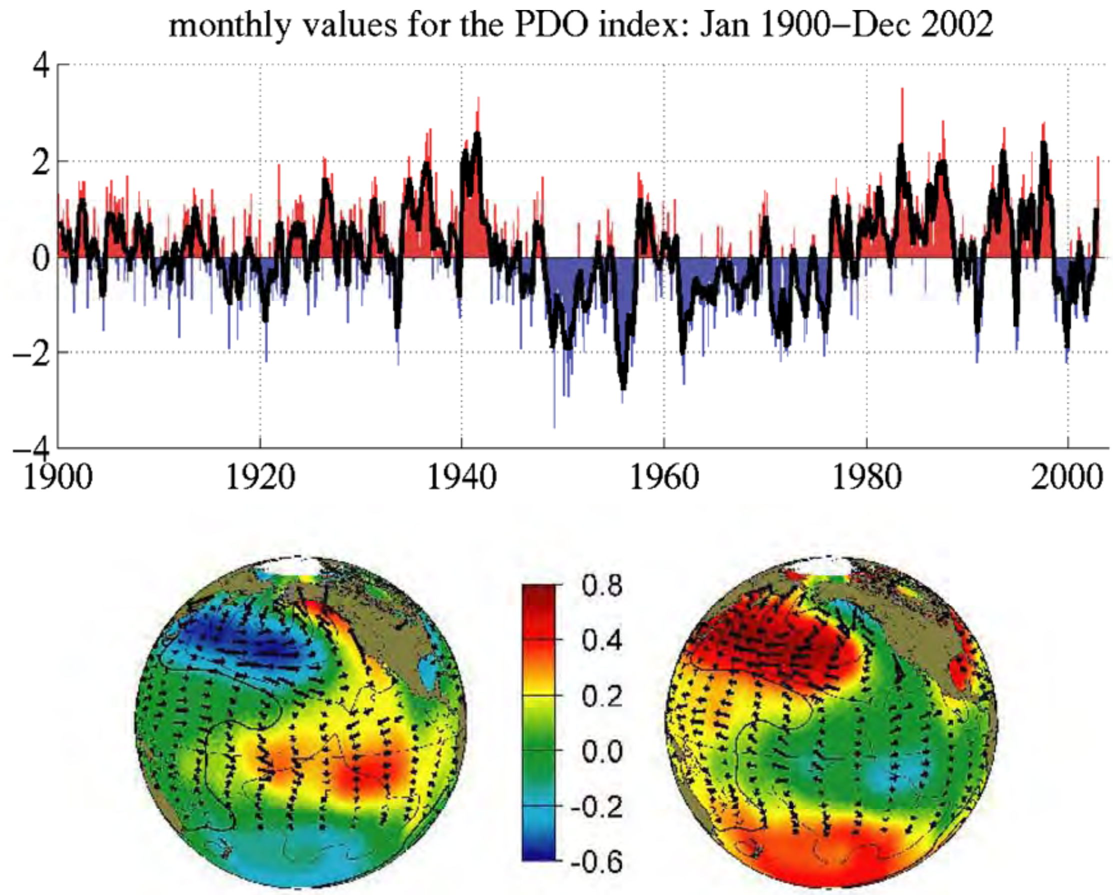


Figure 3. Typical wintertime Sea Surface Temperature (colors), Sea Level Pressure (contours) and surface wind stress (arrows) anomaly patterns during warm and cool phases of PDO. Red colors indicate warm, blue indicates cool.

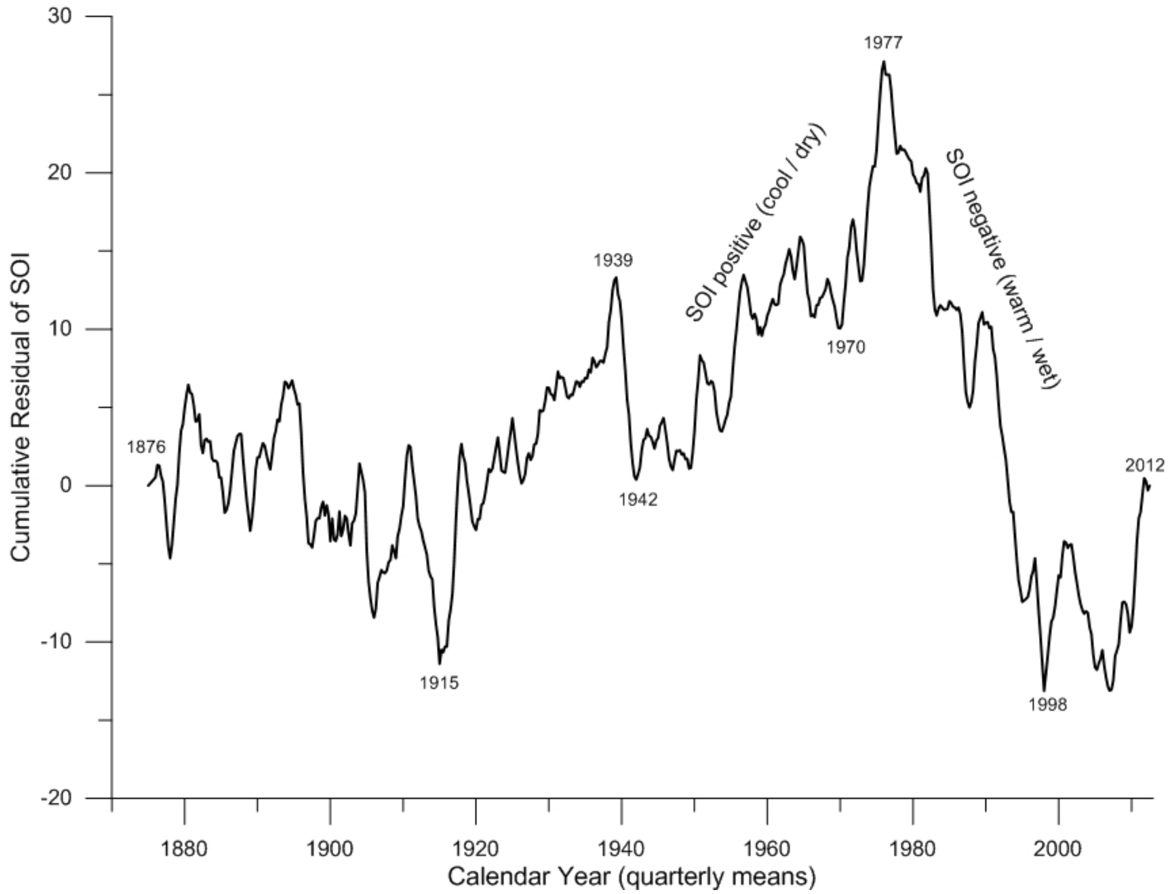


Figure 4. Cumulative residual of quarterly values of Southern Oscillation Index (SOI) [data from Australian Commonwealth Bureau of Meteorology].

Beginning in 1978, the southern California climate began transitioning into a warmer wetter period characterized by a succession of powerful El Niños, particularly those in, 1978, 1980, 1983, 1993, 1995 and 1998, [Inman & Jenkins, 1997]. The average SOI for this period was -0.5, with the 1978/79 El Niño averaging -1.2, the 1982/83 El Niño averaging a record -1.7 and the 1993/94 El Niño recording a mean of -1.0. Heavy rainfall accompanied each of these El Niño events (see Figure 2) causing flood run off to exceed many times the long term mean for all the major rivers and tributaries throughout Southern California (Inman and Jenkins, 1999). The wave climate in southern California also changed, beginning with the El Niño years of 1978/79 and extending until 1998. The prevailing northwesterly winter waves were replaced by high energy waves approaching from the west or southwest, and the previous southern hemisphere swell waves of summer have been replaced by shorter period tropical storm waves during late summer months from the more immediate waters off Central America. Other strong El Niño events of the past have also been accompanied by extreme wave events, although none of these have been as sustained as the succession of El Niños from 1978 to 1995. The 1939/42 El Niño had an average SOI of -1.3 and was associated with a series of destructive wave events in the Southern California Bight, the most intense being the 24/25 September 1939 storm which seriously damaged the breakwater system at Long Beach, CA. The El Niño of 1904/05 had a mean SOI of -1.4 and was attended by a series of damaging west swells in March 1904 and again in March 1905 [Horrer, 1950; Marine Advisors, 1961].

A similar succession of El Niño floods also preceded the cool/dry period of 1944-77, causing major episodes of sediment yield in 1927, 1937, 1938, 1941 and 1943.

From Figures 2 - 4 it is apparent that an inter-decadal pattern of rainfall and SOI has persisted for at least the last century an a half, characterized by alternating cool/dry La Niña dominated periods with little or no sediment yield, followed by warm/wet El Niño dominated periods when heavy rainfall produces most of the total sediment runoff. This kind of inter-decadal climate variability is observed throughout the west coast of the Americas and is now known as the ***Pacific Interdecadal Oscillation (PDO)***, see Mantua et al (1997) and Zhang et al (1997). In this study we attempt to capture the potential range of PDO variability in the discharge problem by constructing the longest possible time series of the seven controlling model inputs from existing data bases, and then invoke statistical searches of those time series for the wet and dry extremes.

3.1 Boundary Conditions

A) Bathymetry: Bathymetry provides a controlling influence on all of the coastal processes at work in both the nearfield and farfield of La Jolla Shores. The bathymetry consists of two parts: 1) a stationary component in the offshore where depths are roughly invariant over time, and 2) a non-stationary component in the nearshore where depth variations do occur over time. The stationary bathymetry generally prevails at depths that exceed ***closure depth*** which is the depth at which net on/offshore transport vanishes. Closure depth is typically -15 m MSL in the Oceanside Littoral Cell, [Inman et al. 1993]. The stationary bathymetry was derived from the National Ocean Survey (NOS) digital database as plotted in Figure 5 seaward of the 15m depth contour. Gridding is by latitude and longitude with a 3 x 3 arc second grid cell resolution yielding a computational domain of 15.4 km x 18.5 km. Grid cell dimensions along the x-axis (longitude) are 77.2 meters and 92.6 meters along the y-axis (latitude). This small amount of grid distortion is converted internally to Cartesian coordinates, using a Mercator projection of the latitude-longitude grid centered on Scripps Pier. The convention for Cartesian coordinates uses x-grid spacings for longitude and y-grid spacings for latitude.

For the non-stationary bathymetry data inshore of closure depth (less than -15 m MSL) we use the equilibrium beach algorithms from Jenkins and Inman (2006). Depth contours generated from these algorithms vary with wave height, period and grain size and are plotted in Figure 5 landward of the 15m depth contour for the wave parameters of the wet weather extreme scenario (see Section 3.3).

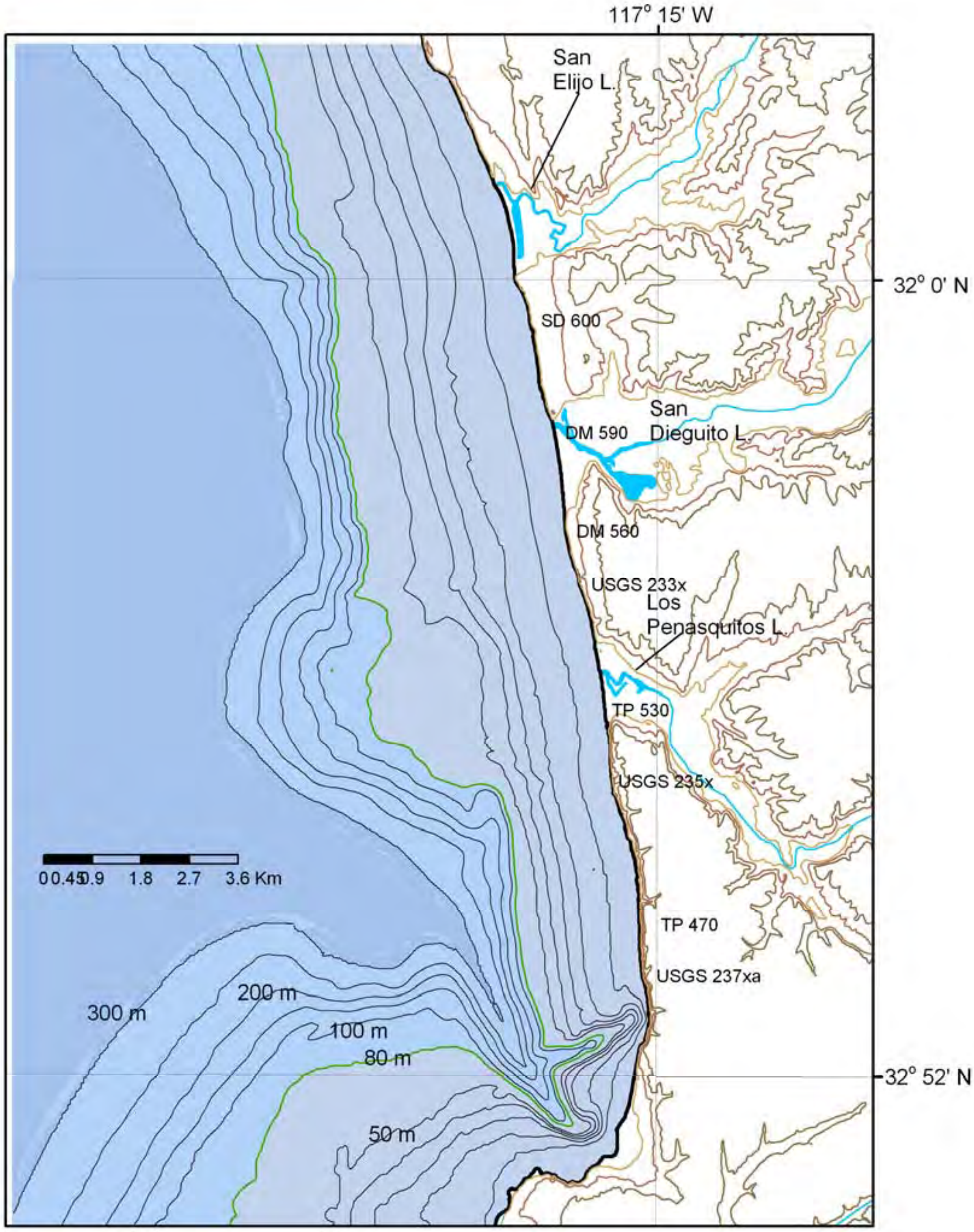


Figure 5. Composite bathymetry from NOS data base and equilibrium profiles after Jenkins and Inman (2006) for wave conditions of wet weather scenario. Depth contours shown in meters mean sea level.

B) Ocean Salinity: Since the three La Jolla Shores outfalls (SDL-157, SDL-062 and SDL-186) discharge predominantly fresh water into a surfzone, the ocean salinity variation used in equation (3) to initialize SEDXPORT is of fundamental importance to the mixing of storm water in the receiving waters. Figure 6a shows the variation in daily mean sea surface salinity in the coastal waters off La Jolla Shores while Figure 7 shows daily mean seafloor salinity, both plotted from 32 years of monitoring data derived from the archival data bases of Scripps Institution of Oceanography (Scripps Pier Shore Station, SIO, 2012) and the Coastal Data Information Program (CDIP, 2012), supplemented by site monitoring data from MBC *Applied Environmental Sciences* (MBC), MBC (2012). The period of these unbroken archival sources extends from 1980 until March 2012. In the period of record from 1980 to 2012, the ocean salinity varies naturally by 10% between summer maximums and winter minimums, with a long term average value of 33.52 parts per thousand (ppt) on the surface and 33.49 ppt on the seafloor. Average sea surface salinity is slightly higher due to evaporation. Maximum salinity was 34.3 ppt on the sea surface and seafloor during the 1998 summer El Nino when southerly winds transported high salinity water from southern Baja up into the Southern California Bight. Minimum salinity was 31.06 ppt on the sea surface and seafloor 30.4 ppt on the seafloor during the 1992 floods. The variation between maximum and minimum salinity is about 3.2 ppt to 3.9 ppt, which is about 10 % of the depth-averaged value of 33.5 ppt. The ocean salinity exceeded the 33.5 ppt average value during 3,736 days during the period of record, and was below average during 2,259 days. Therefore above average salinities are more common than below average salinities. Average salinities were observed a total of 5,022 days of the period of record, or about 46 % of the time. (These data are also confirmed by long term salinity monitoring at Scripps Pier NOAA Station #941-0230, and by 55 CalCOFI cruises in the Southern California Bight between 1984 and 1997, see SIO, 2005; Roemmich, 1989, and Bograd, et al, 2001).

C) Ocean Temperature: The ocean temperature effects the buoyancy of the storm water discharge through the absolute temperature of the discharge. This buoyancy effect is calculated by the specific volume change of the discharge relative to the ambient ocean water. The buoyancy of the plume exerts a strong effect on the mixing and rate of assimilation of the sea salts and backwash constituents by the receiving waters.

We use the average of temperature records from the archival data bases of Scripps Institution of Oceanography (Scripps Pier Shore Station, SIO, 2012) and the Coastal Data Information Program (CDIP, 2012), supplemented by site monitoring data from MBC (2012). An 11,688 point record of daily mean sea surface temperatures are plotted in Figure 6b, while daily mean seafloor temperatures are plotted in Figure 7b. These temperature data were throughput to dilution model as detailed in Section 2. A pronounced seasonal variation in these temperatures is quite evident with the maximum recorded daily mean temperature reaching 25.4 ^{o}C on the sea surface and 24.4 ^{o}C on the seafloor during the summer of the 1993 El Niño; and the minimum falling to 9.9 ^{o}C on the sea surface and 11.0 ^{o}C on the seafloor during the winter of the 1999-2000 La Niña. The mean temperature was found to be 17.7 ^{o}C on the sea surface and 17.2 ^{o}C on the seafloor. On a percentage basis, the natural variability of the temperature of coastal waters of La Jolla Bay is significantly greater than that of salinity, where temperature variability is on the order of $\Delta T = 86\%$ vs salinity variability of $\Delta S = 10\%$.

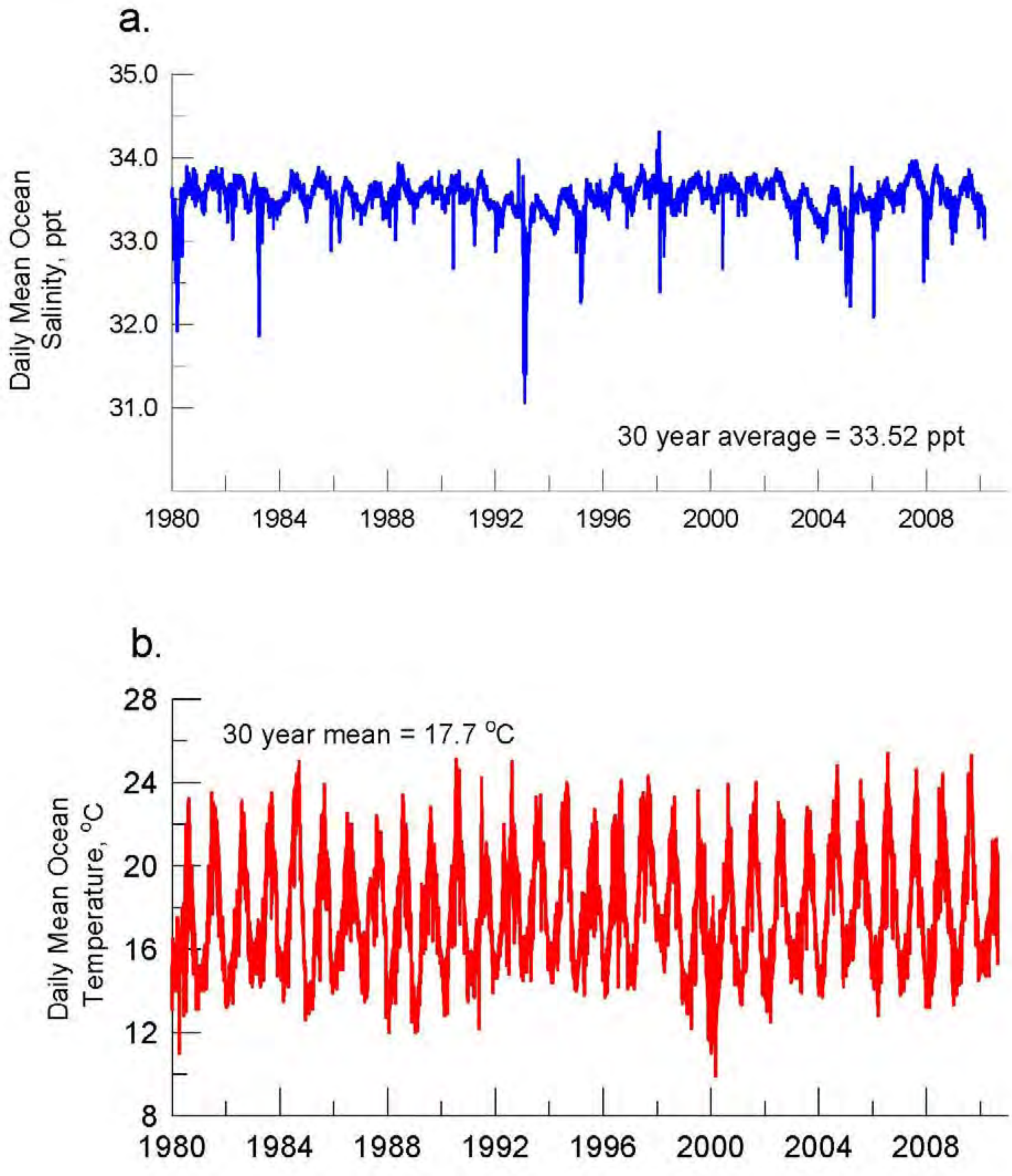


Figure 6. Daily mean ocean surface salinity (a) and daily mean ocean surface temperature at La Jolla, CA; from CDIP (2012), SIO, (2010), and MBC (2012).

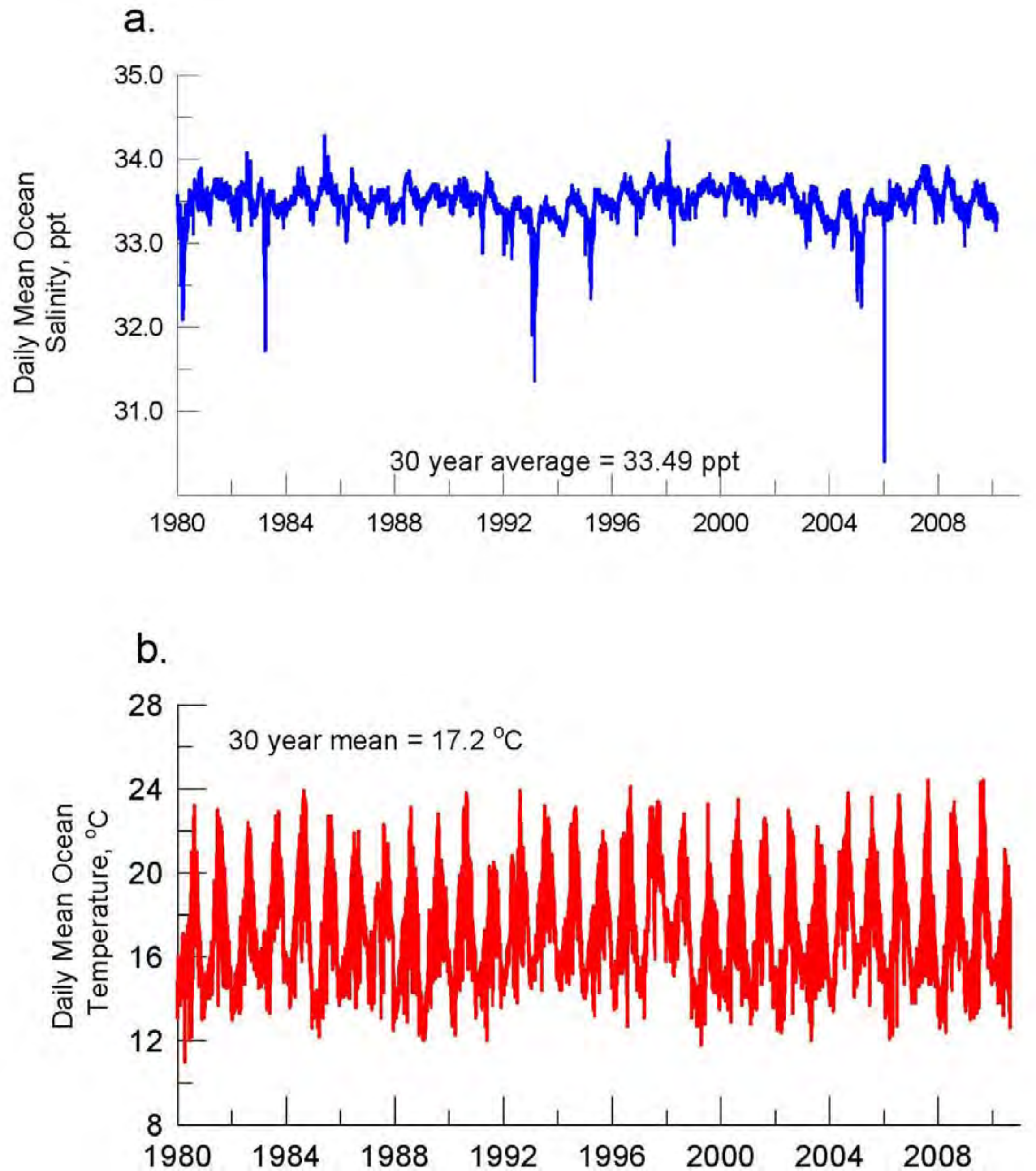


Figure 7. Daily mean ocean surface salinity (a) and daily mean ocean bottom temperature at La Jolla, CA; from CDIP (2012), SIO, (2010), and MBC (2012).

D) Ocean Water Levels: In the shallow nearshore environment off La Jolla Shores Beach, the dilution volume is limited by the time variation in ocean water level. Ocean water level varies in response to both tidal and climate oscillations such as ENSO. The ocean water level is monitored by the tide gage station located on Scripps Pier, La Jolla, CA. This tide gage (NOAA #941-0230) was last leveled using the 1983-2001 tidal epoch. Elevations of tidal datums referred to Mean Lower Low Water (MLLW), in **METERS** are as follows:

HIGHEST OBSERVED WATER LEVEL (11/13/1997) = 2.332 m
MEAN HIGHER HIGH WATER (MHHW) = 1.624 m
MEAN HIGH WATER (MHW) = 1.402 m
MEAN TIDE LEVEL (MTL) = 0.839 m
MEAN SEA LEVEL (MSL) = 0.833 m
MEAN LOW WATER (MLW) = 0.276 m
NORTH AMERICAN VERTICAL DATUM-1988 (NAVD) = 0.058 m
NGVD29 = 0.700 m
MEAN LOWER LOW WATER (MLLW) = 0.000 m
LOWEST OBSERVED WATER LEVEL (12/17/1933) = -0.874 m

Water levels measured by the Scripps Pier Tide Gage (#941-0230) have been archived by NOAA (2005) for the period of record, 1980 to 2012. Reconstruction of a water level time series was performed on the entire set of 1980-2012 NOAA measurements. The resulting time series of daily maximum and minimum ocean water levels is plotted in Panel-b of Figure 8. The positive sea level anomalies are a persistent and sustained occurrence in the observations of ocean water levels during this period of record and are another signature of El Niño. The warming of the coastal ocean during El Niño events causes thermal expansion of seawater by the second term in the equation of state, equation (3). A very significant number of diurnal tide cycles during the 32 period of record (466) have produced water level elevations well in excess of the extreme higher-high water levels (EHHW) of the astronomic tides, (where EHHW = +4.28 ft. NGVD for a perigean spring tide occurring once every 4.5 years). In the period of record shown by the blue trace in Figure 8b, the maximum ocean water level was +5.35 ft. NGVD occurring during the 1997 El Niño, 1.31 ft. higher than the astronomic tides of the tide tables. These high water levels promote initial dilution of any beach discharge because they provide additional water depth and dilution volume where these discharges enter the sea. On the other hand, the minimum ocean water level was -4.66 ft. NGVD, occurring during the 1988 winter. These low water levels shown in the green trace of Figure 8b reduce initial dilution of beach discharge.

E) Discharge Flow Rates and Constituent Mass Loads: Wet weather monitoring was performed during three storm events during the 2011-2012 wet weather monitoring season, (AMEC, 2012). Table 2 presents the dates, sites monitored, and total rainfall for each of the three significant monitored events. Appendix G gives tabular listings of the AMEC monitoring data.

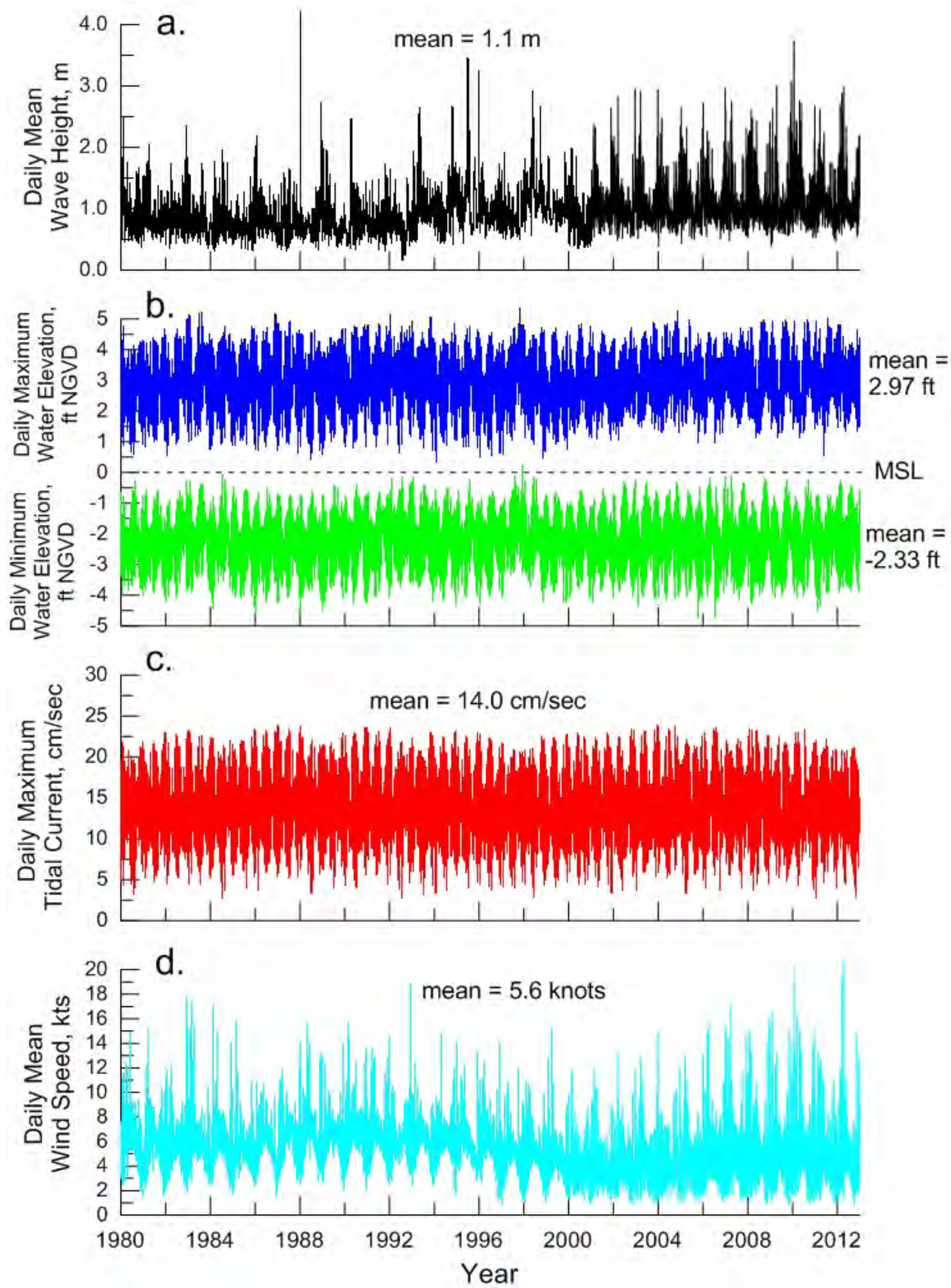


Figure 8. Controlling ocean forcing functions for La Jolla Bay: a) daily mean significant wave height, b) daily high and low water levels, c) daily maximum tidal current, d) daily mean wind speed. (data from CDIP, 2012; SIO, 2012; and NCEP, 2012).

**Table 2.
Monitoring Events Summary**

Event	Date	Total Rainfall (inches)	Sites Monitored
Event 1	11/20 to 21/2012	0.89	SDL-062-OD/RW SDL-157-OD/RW
Event 2	2/7/2012	0.23	SDL-062-OD/RW SDL-157-OD/RW SDL-186-OD/RW
Event 3	3/17 to 18/2012	0.54	SDL-062-OD/RW/SED SDL-157-OD/RW/SED SDL-186-OD/RW/SED

Notes:
OD - Outfall Discharge
RW - Receiving Water
SED - Sediment

Per the Special Protections Document Core Discharge Monitoring program, storm water outfalls 18 inches in diameter or greater must be monitored. Of the four outfalls within the La Jolla Shores Coastal Watershed that meet these criteria, the two largest outfalls (SDL-062 and SDL-157) were monitored during three events listed in Table 2. During these wet weather events, flow-weighted composites were collected, in addition to time-weighted composites composed of two to four grab samples that were collected during two storm events at a third outfall monitoring location (SDL-186). (The remaining two outfalls (SDL-063 and SDL-063B) were not sampled). The concentrations in mass per unit volume $\rho_{0i}(t)$ that were measured at time t for constituent i were applied to the measured flow rates $Q_j(t)$ of outfall j to obtain the flux $J_{ji}(t)$ of that constituent from outfall j, or $J_{ji}(t) = \rho_{0i}(t)Q_j(t)$. The cumulative mass loading $M_{ji}(\Delta t)$ of constituent i from outfall j over time period Δt is simply the time integral of the fluxes $M_{ji}(\Delta t) = \int_{\Delta t} J_{ji}(t)dt$. These data are used to define the variation the wet conditions that the La Jolla Shores storm drain discharges exert on the receiving water and ASBS 29. These data are also used to identify the maximum discharge rates and concentration of discharge constituents, with primary emphasis here on total suspended solids (TSS) and the cumulative mass loading of total suspended solids, $M_{ji}(\Delta t)$. The dilution modeling of these point sources presented herein does not consider combined effects from non-point source runoff, such as from Peñasquitos Canyon.

Wet Weather Event 1: Rainfall began at 5:01 pm on November 20, 2011 and ended at 12:21 am on November 21, 2011, totaling 0.89 inches. Flow began shortly after the onset of rainfall at both monitored locations, SDL-062 and SDL-157. Figure 9 presents the hydrographs for storm drains SDL-062 (black) and SDL- 157 (blue) as compared against rainfall (red) during Wet Weather Event-1. Peak flow rates were for storm drain SDL-157 were 3.5 cfs with a storm total flow volume of 53,425 cubic feet. Peak flow rates were on the order of 11 cubic feet per second (cfs) for storm drain SDL-062 while the storm total flow volume was 120,569 cubic feet. Figure 10 gives the flux of total suspended solids in a 15 minute interval (red) and cumulative mass loading of total suspended solids (black) for storm drain SDL-157 during Wet Weather Event-1. Peak TSS fluxes for SDL-157 were 28 kg in a 15 minute interval, or 1.9 kg/min; and the cumulative TSS mass loading from this storm drain for the entire event was 515 kg. Figure 11 gives the corresponding TSS fluxes and cumulative mass loading from storm drain SDL-062 for Wet Weather Event-1. Peak TSS fluxes for SDL-062 were significantly higher, on the order of 70 kg in a 15 minute interval, or 4.7 kg/min; and the cumulative TSS mass loading from SDL-062 gave a storm total of 888 kg. This is the highest TSS total load measured during any event or for any storm drain during the entire 2011-2012 monitoring period, as a consequence of Wet Weather Event-1 having the highest rainfall totals and representing a first-flush type of event.

Wet Weather Event 2: Rainfall began at 3:06 pm on February 7, 2012 and ended at 5:46 pm on February 7, 2012, totaling 0.23 inches. Flow began shortly after the onset of rainfall at the three monitored locations, SDL-062, SDL-157, and SDL-186. Figure 12 presents the hydrographs for storm drains SDL-062 (black) and SDL- 157 (blue) as compared against rainfall (red) during Wet Weather Event-2. Peak flow rates were for storm drain SDL-157 were 3.5 cfs with a storm total flow volume of 25,248 cubic feet. Peak flow rates were on the order of 7 cubic feet per second (cfs) for storm drain SDL-062 while the storm total flow volume was 27,052 cubic feet. Figure 13 gives the flux of total suspended solids in a 15 minute interval (red) and cumulative mass loading of total suspended solids (black) for storm drain SDL-157 during Wet Weather Event-2. Peak TSS fluxes for SDL-157 were 20 kg in a 15 minute interval, or 1.3 kg/min; and the cumulative TSS mass loading from this storm drain for the entire event was 164 kg. Figure 14 gives the corresponding TSS fluxes and cumulative mass loading from storm drain SDL-062 for Wet Weather Event-1. Peak TSS fluxes for SDL-062 were slightly greater, on the order of 22 kg in a 15 minute interval, or 1.4 kg/min; but the cumulative TSS mass loading from SDL-062 was less, about 100 kg, due to a rapid decline in TSS fluxes during the latter portion of the storm hydrograph for SDL-062 during Wet Weather Event-2.

Wet Weather Event 3: Rainfall began at 3:55 am on March 17, 2012 and ended at 7:31 pm on March 18, 2012, totaling 0.54 inches. Flow began shortly after the onset of rainfall at the three monitored locations, SDL-062, SDL-157, and SDL-186. Figure 15 presents the hydrographs for storm drains SDL-062 (black) and SDL- 157 (blue) as compared against rainfall (red) during Wet Weather Event-3. Peak flow rates were for storm drain SDL-157 were about 3.0 cfs with a storm total flow volume of 49,414 cubic feet. Peak flow rates were on the order of 7 cubic feet per second (cfs) for storm drain SDL-062 while the storm total flow volume was 89,485 cubic feet. Figure 16 gives the flux of total suspended solids in a 15 minute interval (red) and cumulative mass loading of total suspended solids (black) for storm drain SDL-157 during Wet Weather Event-3. Peak TSS fluxes for SDL-157 were 13 kg in a 15 minute interval, or 0.9 kg/min; and the cumulative TSS mass loading from this storm drain for the entire event was 238 kg. Figure 17 gives the corresponding TSS fluxes and cumulative mass loading from storm drain SDL-062 for Wet Weather Event-3. Peak TSS fluxes for SDL-062 were slightly greater, on the order of 15 kg in a 15 minute interval, or 1.0 kg/min; but the cumulative TSS mass loading from SDL-062 was less, about 205 kg; again due to a rapid decline in TSS fluxes during the latter portion of the storm hydrograph for SDL-062 during Wet Weather Event-3.

Discharge from storm drain SDL-186 was not measured during the monitoring program, but was modeled in AMEC (2012). For the purposes of initializing SDL-186 in the SEDXPORT model in the present study, we use these modeled discharges. For Wet Weather Event-1, flow volume for SDL-186 was taken as 28,237 cubic feet. For Wet Weather Event-2, flow volume for SDL-186 was taken as 4,525 cubic feet. For Wet Weather Event-3, flow volume for SDL-186 was taken as 12,137 cubic feet.

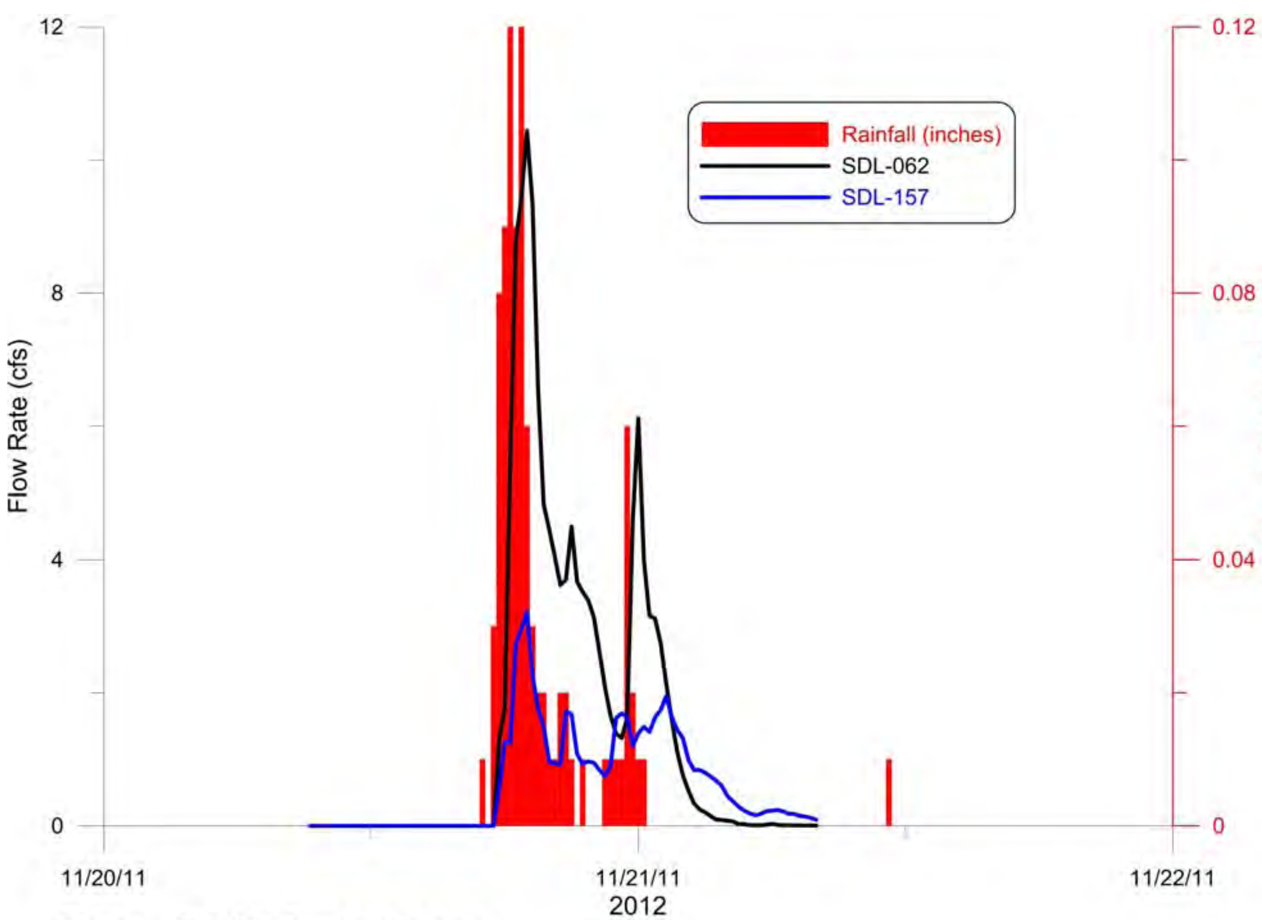


Figure 9. Hydrographs for storm drains SDL-062 (black) and SDL- 157 (blue) as compared against rainfall (red) during Wet Weather Event-1 , 20 November 2011 to 21 November 2011.

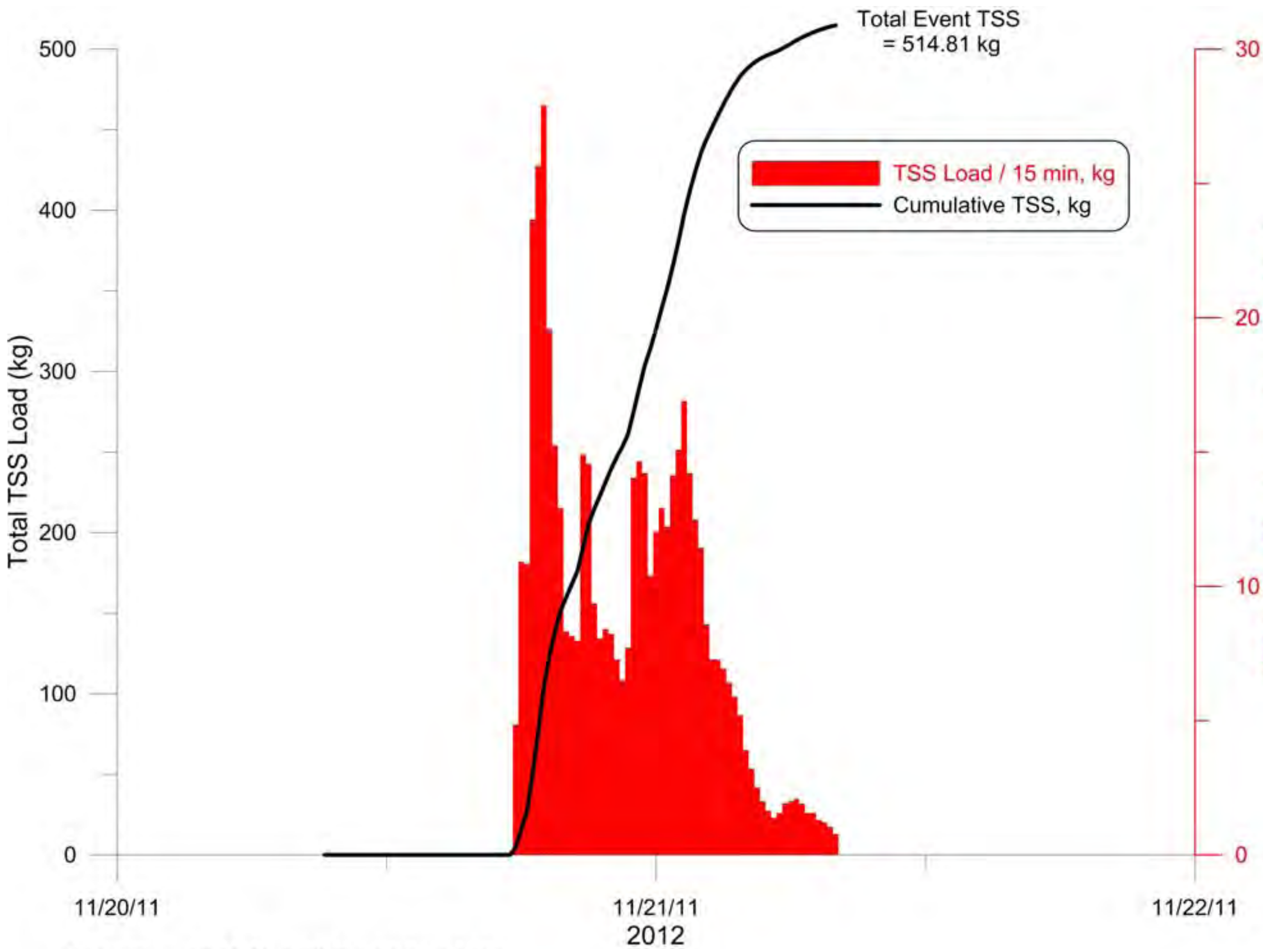


Figure 10. Flux of Total Suspended Solids in a 15 minute interval (red) and Cumulative Mass Loading of Total Suspended Solids (black) for storm drain SDL-157 during Wet Weather Event-1 , 20 November 2011 to 21 November 2011.

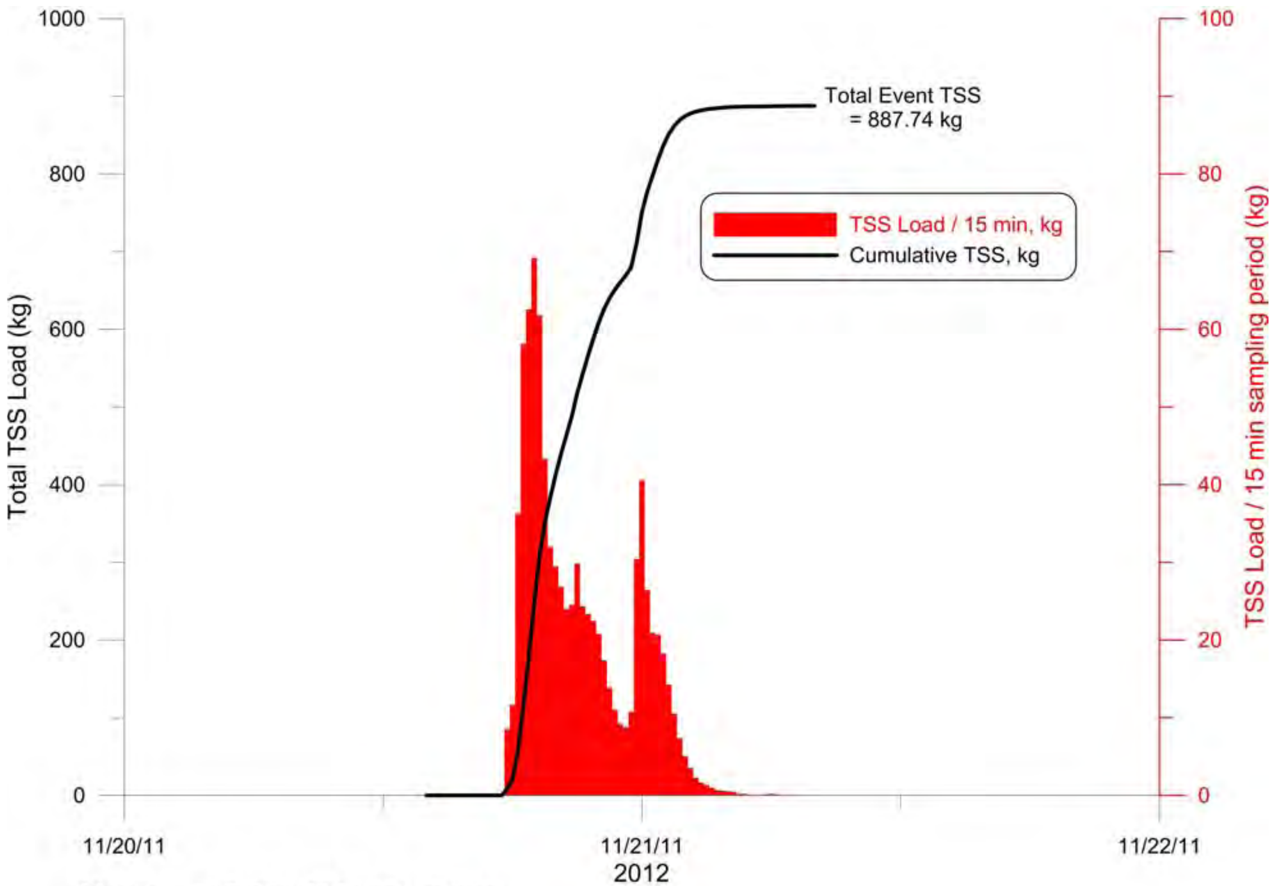


Figure 11. Flux of Total Suspended Solids in a 15 minute interval (red) and Cumulative Mass Loading of Total Suspended Solids (black) for storm drain SDL-062 during Wet Weather Event-1 , 20 November 2011 to 21 November 2011.

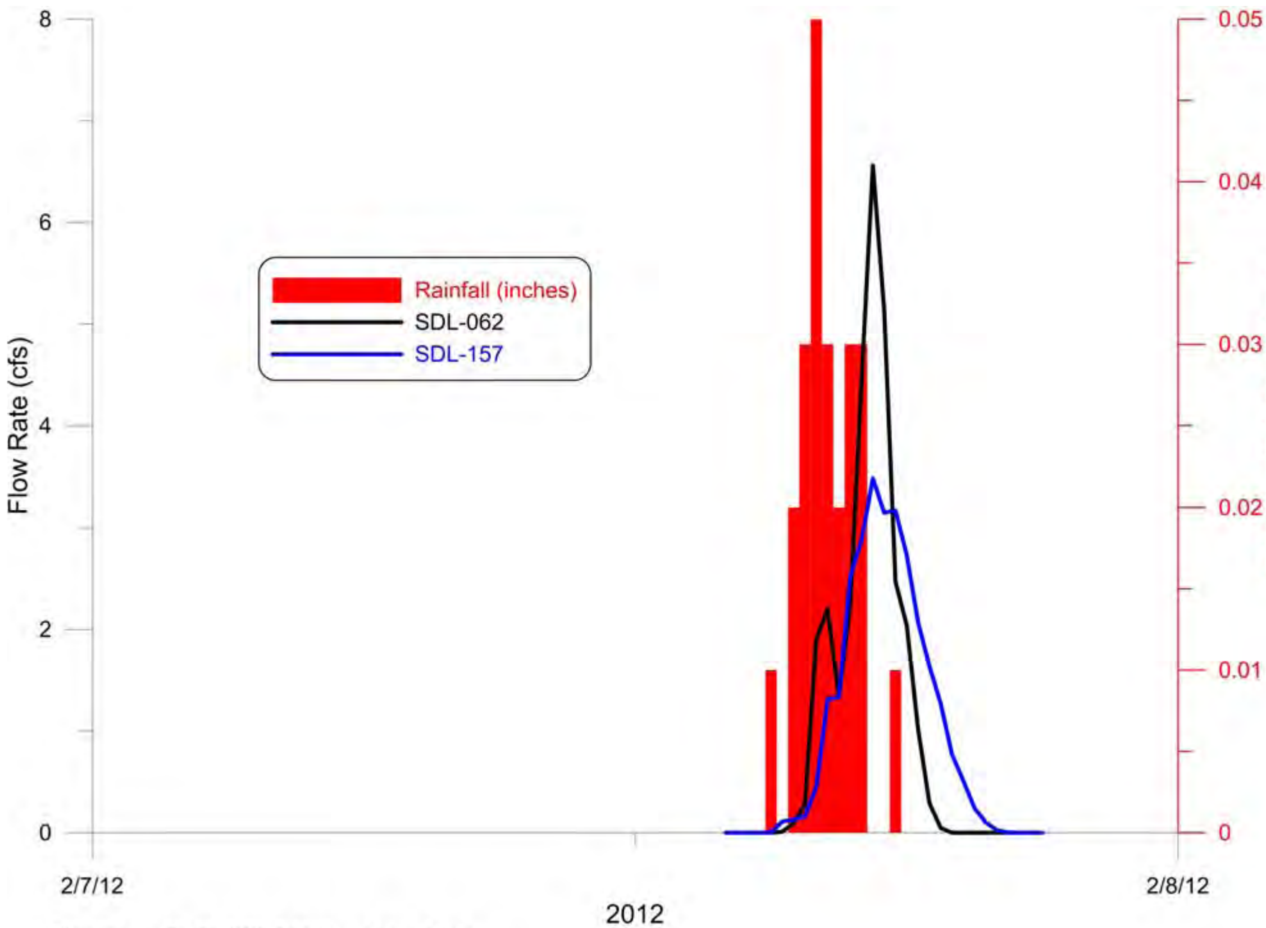


Figure 12. Hydrographs for storm drains SDL-062 (black) and SDL- 157 (blue) as compared against rainfall (red) during Wet Weather Event-2 , 7 February 2012.

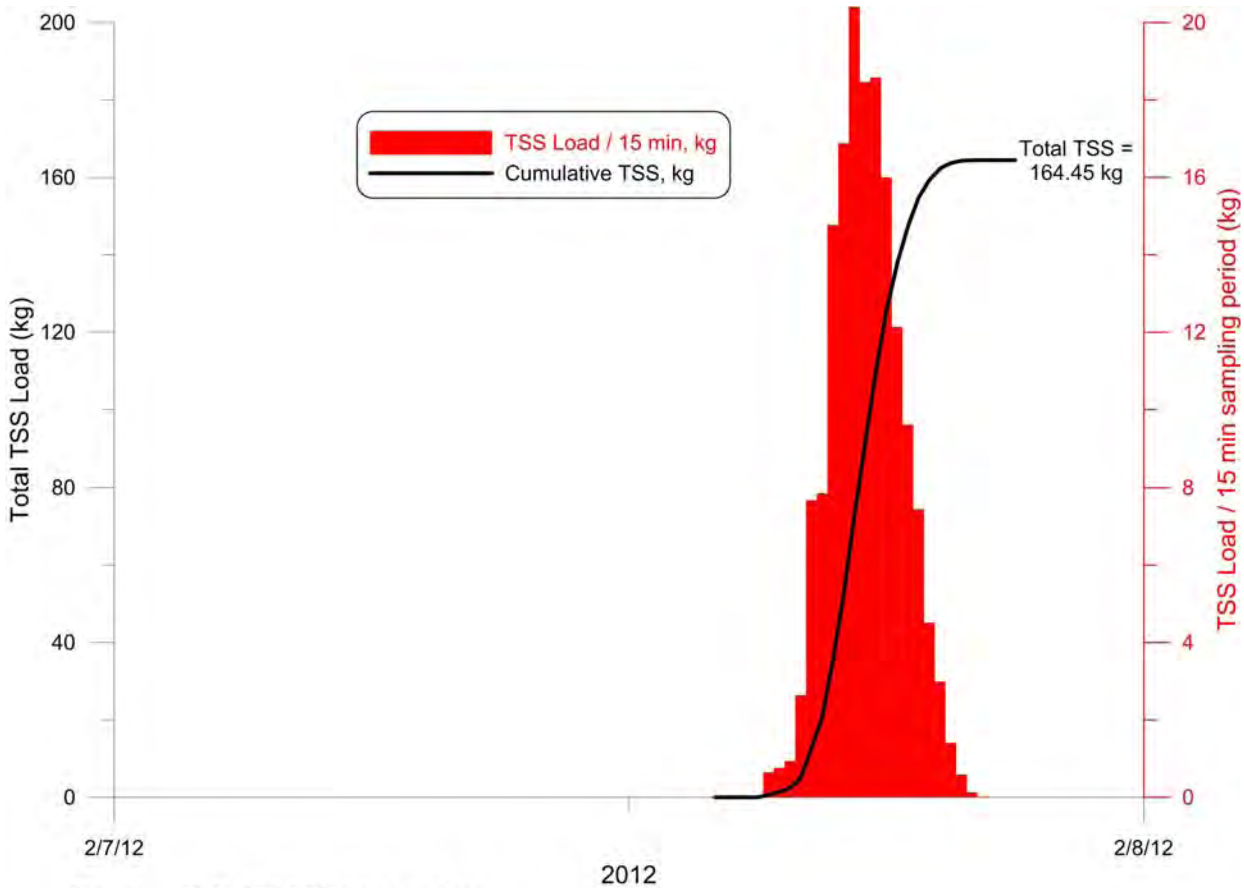


Figure 13. Flux of Total Suspended Solids in a 15 minute interval (red) and Cumulative Mass Loading of Total Suspended Solids (black) for storm drain SDL-157 during Wet Weather Event-2 , 7 February 2012.

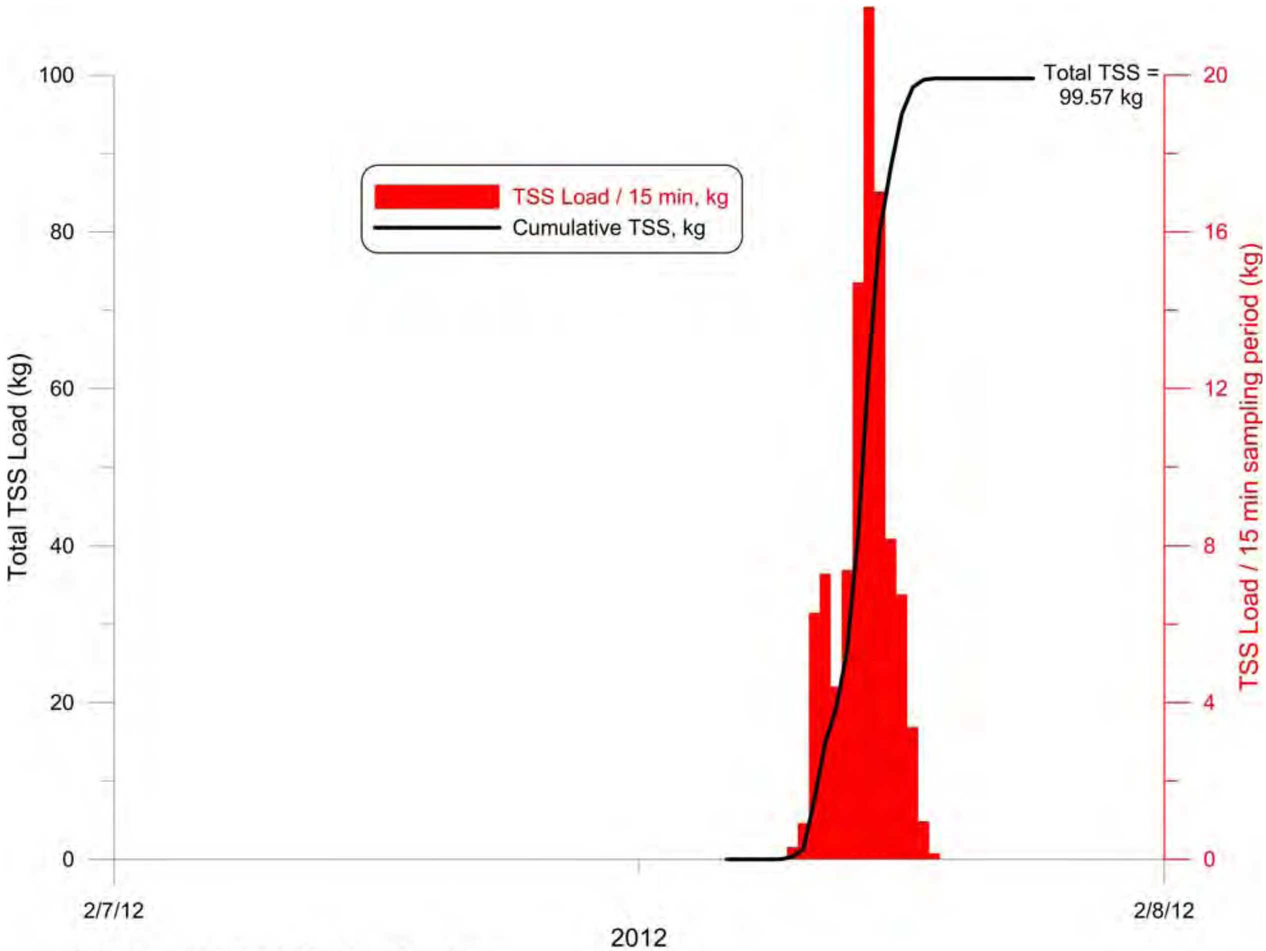


Figure 14. Flux of Total Suspended Solids in a 15 minute interval (red) and Cumulative Mass Loading of Total Suspended Solids (black) for storm drain SDL-062 during Wet Weather Event-2 , 7 February 2012.

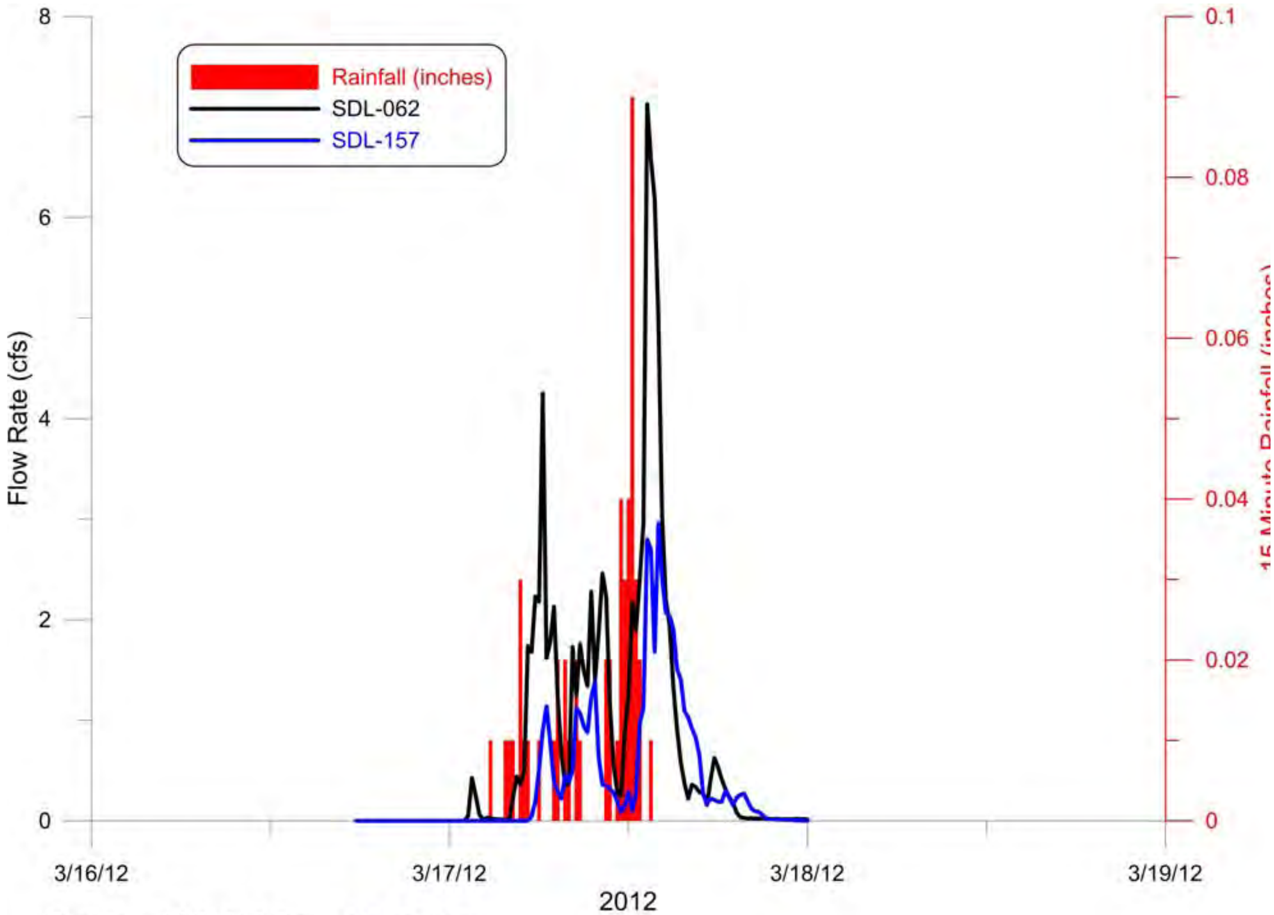


Figure 15. Hydrographs for storm drains SDL-062 (black) and SDL- 157 (blue) as compared against rainfall (red) during Wet Weather Event-3 , 17 March 2012 to 18 March 2012.

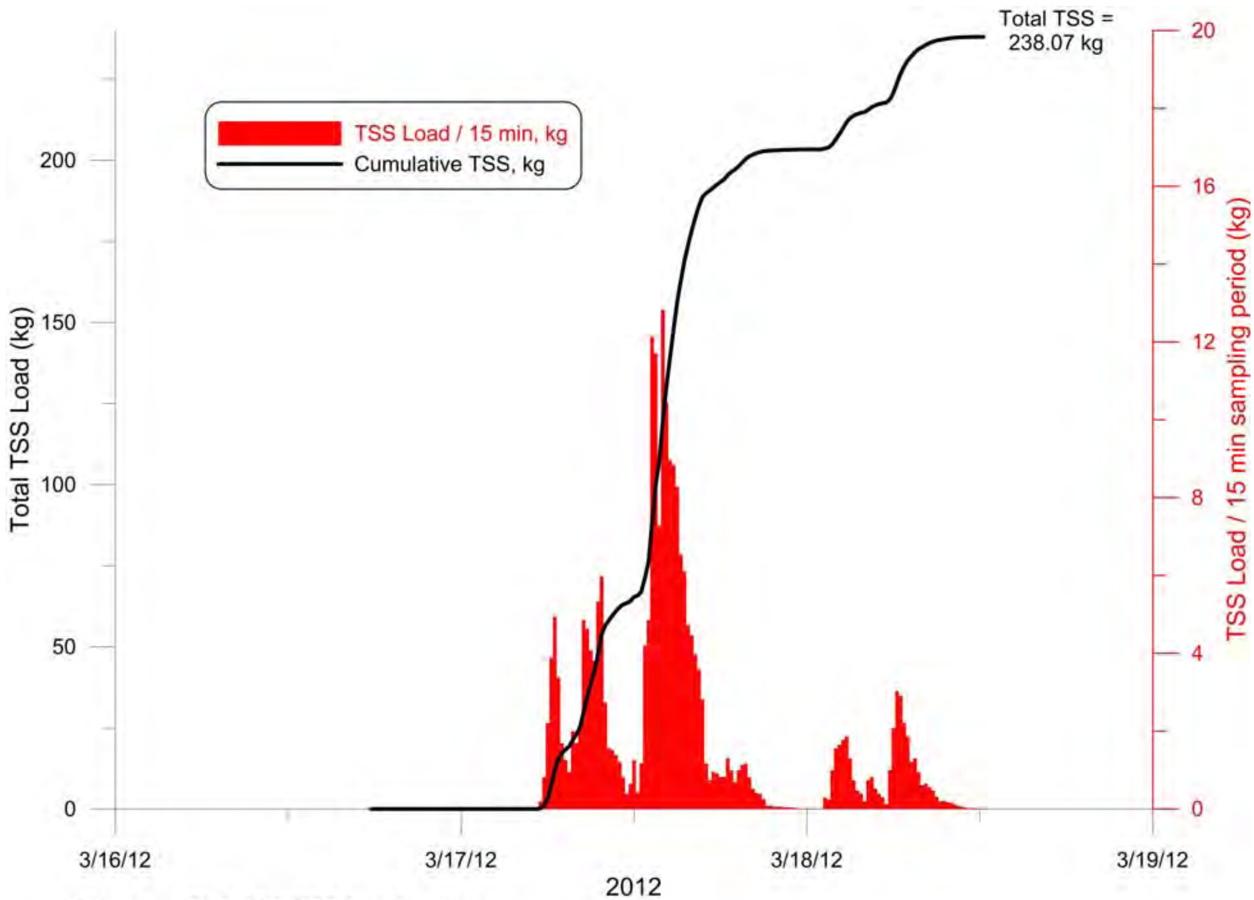


Figure 16. Flux of Total Suspended Solids in a 15 minute interval (red) and Cumulative Mass Loading of Total Suspended Solids (black) for storm drain SDL-157 during Wet Weather Event-3 , 17 March 2012 to 18 March 2012.

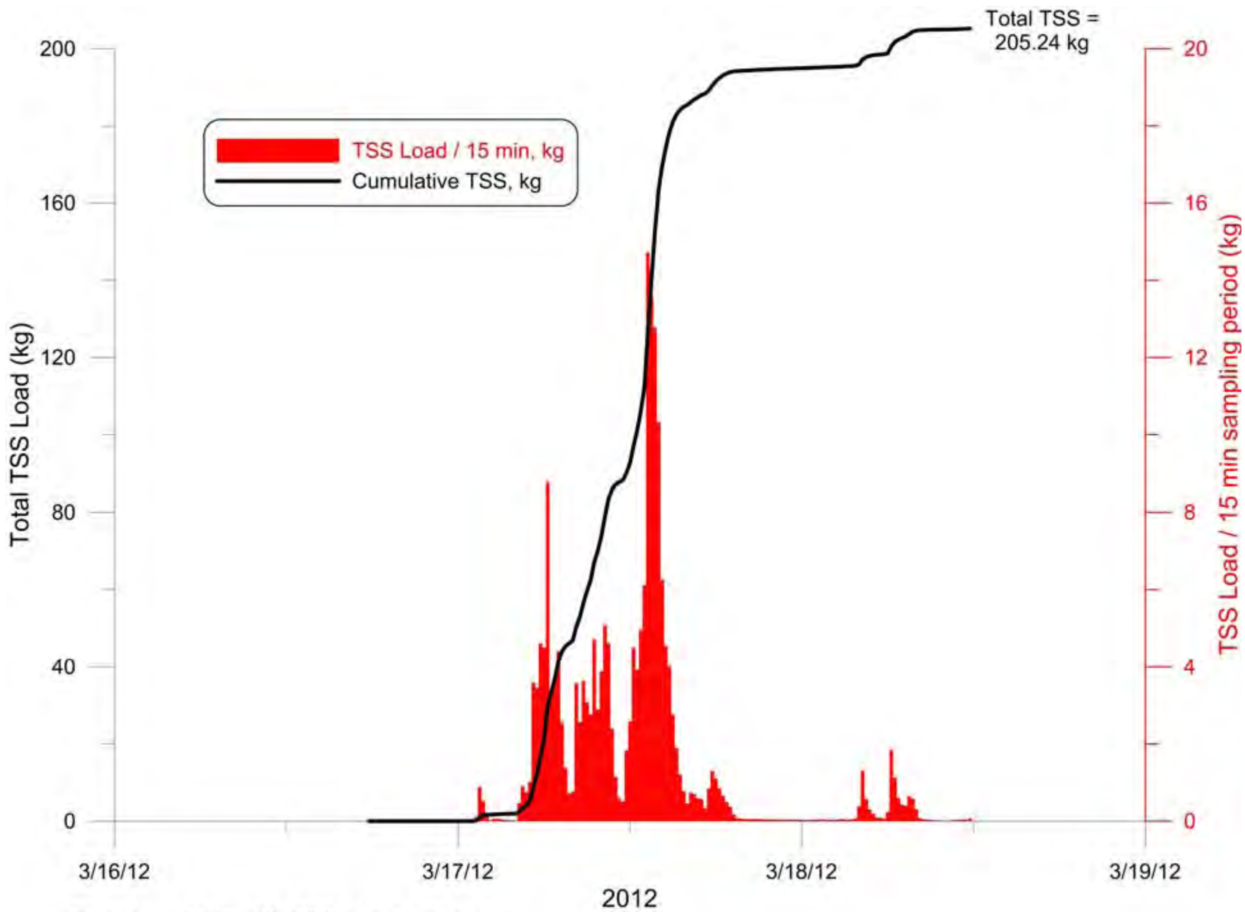


Figure 17. Flux of Total Suspended Solids in a 15 minute interval (red) and Cumulative Mass Loading of Total Suspended Solids (black) for storm drain SDL-062 during Wet Weather Event-3 , 17 March 2012 to 18 March 2012.

3.2 Forcing Functions

A) Waves: Because the combined discharges of storm water from the La Jolla Shores storm drains are discharged into the surfzone, the wave climate exerts leading order control on the initial dilution and dispersion of TSS as well as the copper or TCDD constituents adsorbed on the surfaces of the fine grained sediments that make up the suspended solids loading. The wave forcing records are derived from measurements during the Coastal Data Information Program, (CDIP). This program routinely monitored waves at several locations in the lower Southern California Bight since 1980. The nearest CDIP *directional* wave monitoring sites for the Oceanside Littoral Cell and Torrey Pines Sub-Cell (Figure 18) are:

a. Oceanside Array

- Station ID: 00401
- Location:

- 33 11.4^0 North, 117 23.4^0 West
- 500 feet SW of pier

• Water Depth (m): 10
• Instrument Description:
  - Underwater Directional Array
• Measured Parameters:
  - Wave Energy
  - Wave Direction

b. San Clemente

• Station ID: 05201
• Location:
  - 33 25.2^0 North, 117 37.8^0 West
  - 1000 ft NW of San Clemente Pier
• Water Depth (m): 10
• Instrument Description:
  - Underwater Directional Array

c. Measured Parameters:

- Wave Energy
- Wave Direction

d. Huntington Beach Array

• Station ID: 07201
• Location:
  - 33 37.9^0 North, 117 58.7^0 West
  - Approximately 1 mile west of lifeguard headquarters at Huntington Beach, CA
• Water Depth (m): 10
• Instrument Description:
  - Underwater Directional Array
• Measured Parameters:
  - Wave Energy
  - Wave Direction

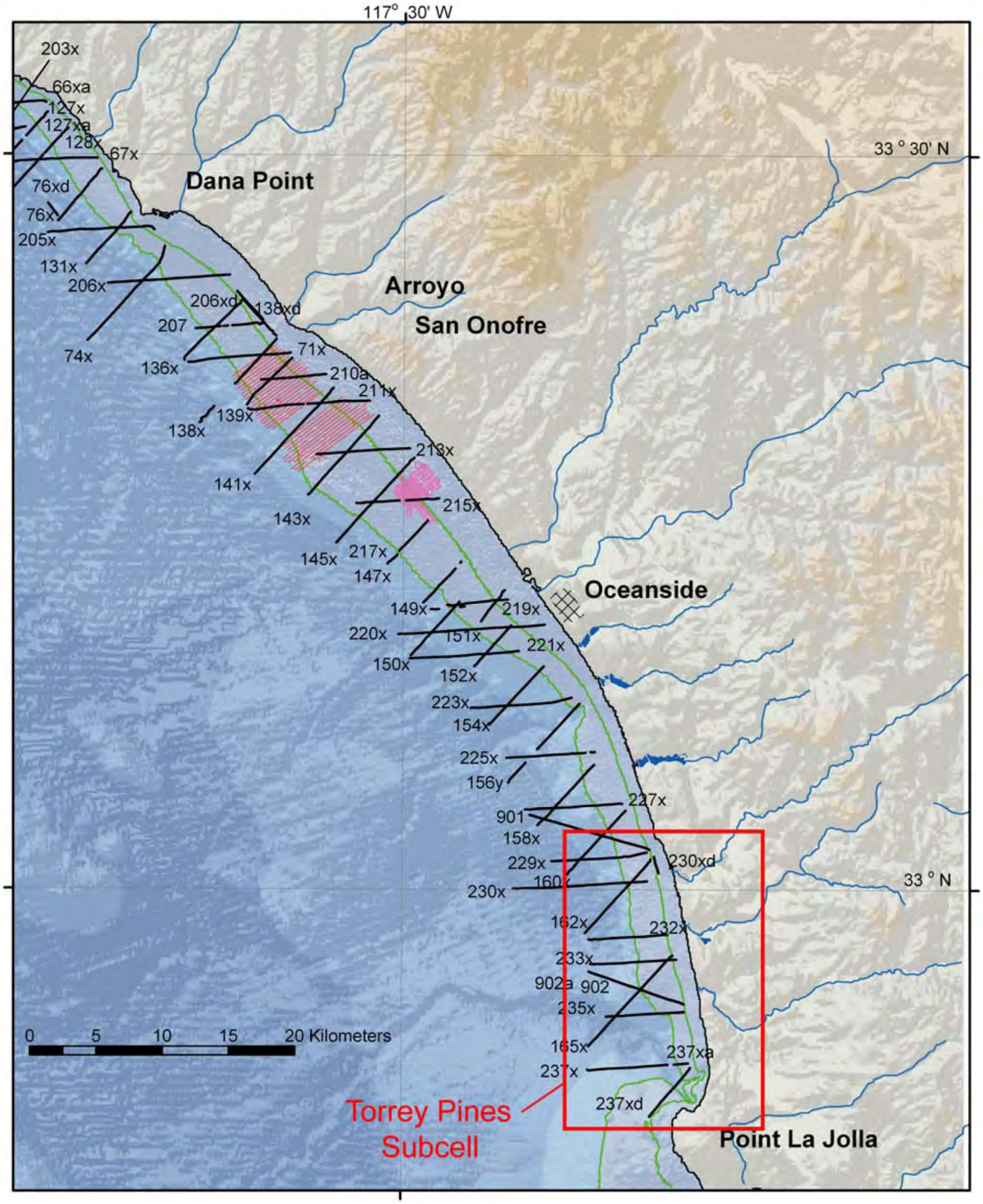


Figure 18. Oceanside Littoral Cell and Torrey Pines Sub-Cell.

Data for these CDIP wave monitoring sites is available beginning January 1980 through March 2012 [CDIP, 2012]. In addition to the CDIP sites, waves have been monitored at Torrey Pines Beach from 1972 until 1984 by the SAS Stations deployed by Scripps Institution of Oceanography, (SIO), Pawka (1982). The ensemble of data from the CDIP and SAS monitoring stations were pieced together into a continuous record from 1980-2012 and entered into a structured preliminary data file.

The data in the preliminary file represent partially shoaled wave data specific to the local bathymetry around Oceanside, San Clemente or Huntington Beach. To correct these data to the Torrey Pine Sub-Cell in which the dilution modeling is performed, the data are entered into a refraction/diffraction numerical code, back-refracted out into deep water, and subsequently brought onshore into the immediate neighborhood of the Torrey Pines Sub Cell (as delineated in Figure 5 and 18). CDIP wave data are shoaled into Scripps Beach and the neighboring beaches of the Torrey Pines Sub-Cell using the **OCEANRDS** refraction-diffraction computer codes. The primitive equations for this code are lengthy, listings of the FORTRAN codes of **OCEANRDS** appear in Appendix C. These codes calculate the simultaneous refraction and diffraction patterns propagating over a Cartesian depth grid. "OCEANRDS" uses the parabolic equation method (PEM), Radder (1979), applied to the mild-slope equation, Berkhoff (1972). To account for very wide-angle refraction and diffraction relative to the principle wave direction, "OCEANRDS" also incorporates the high order PEM Pade approximate corrections modified from those developed by Kirby (1986a-c). Unlike the recently developed REF/DIF model due to Dalrymple et al (1984), the Pade approximates in "OCEANRDS" are written in tesseral harmonics, per Jenkins and Inman (1985); in some instances improving resolution of diffraction patterns associated with steep, highly variable bathymetry along the shelf break. These refinements allow calculation of the evolution and propagation of directional modes from a single incident wave direction; which is a distinct advantage over the more conventional directionally integrated ray methods that are prone to caustics (crossing rays) and other singularities in the solution domain where bathymetry varies rapidly over several wavelengths.

An example of a reconstruction of the back-refracted wave field throughout the Bight is shown in Figure 19 using the CDIP data from the San Clemente array. Wave heights are contoured in meters according to the color bar scale and represent 6 hour averages, not an instantaneous snapshot of the sea surface elevation. Note how the sheltering effects of Catalina and San Clemente Islands have induced longshore variations in wave height throughout the Southern California Bight. These variations (referred to as shadows and bright spots) induce longshore transport away from areas of high waves (bright spots, red) and toward areas of low waves (shadows, dark blue). Figure 20 shows the deep water significant wave heights, periods and directions resulting from the series of back-refraction calculations for the complete CDIP and SIO data set at $\Delta t = 6$ hour intervals over the 1980-2012 period of record. The data in Figure 20 are the values used as the deep water boundary conditions of the forward refraction computations into the Torrey Pines Sub Cell (Figure 18). The deep water wave angles are plotted with respect to the direction (relative to true north) from which the waves are propagating at the deep water boundary of Figure 18. Inspection of Figure 20 reveals that a number of large swells lined up with the wave windows open to the Torrey Pine Sub-Cell during the El Niño's of 1980-83, 1986-

88, 1992-95, and 1997-98. The largest of these swell events was the 18 January 1988 storm, producing 4.5 m deep water swells off Scripps Beach (see event #6 in Figure 20).

Figure 21 gives an example of the forward refraction calculation over the Torrey Pines Sub-Cell and La Jolla Bay region for the low energy waves that characterize the low mixing conditions of the dry weather modeling scenario. These particular waves were observed on 22 August 2011, and had a daily mean wave height of only 0.2 m, approaching La Jolla Shores from 210^0 with 10 sec period. In contrast, the refraction/diffraction pattern for a wet weather scenario is shown in Figure 22 for 1.8 m high storm swells shoaling onto La Jolla Shores from 283^0 with 14 sec period during 21 November 2011 (Wet-Weather Event-1). The longer period northwest swells of the stormy wet weather scenario produce a pronounced pattern of shadows (regions of locally smaller waves) and bright spots (regions of locally higher waves). Wave driven nearshore currents flow away from bright spots and converge on shadows. Inspection of Figure 22 reveals that the Avenedia de La Playa storm drain SDL-062 is in a shadow zone flanked by bright spots to the north near storm drain SDL-157. Consequently the zone of initial dilution for the SDL-062 storm drain is located in a region of converging longshore currents during the wet weather scenario. Such a convergence of drift results in rip currents and offshore flow, acting to disperse the beach discharges into deep water.

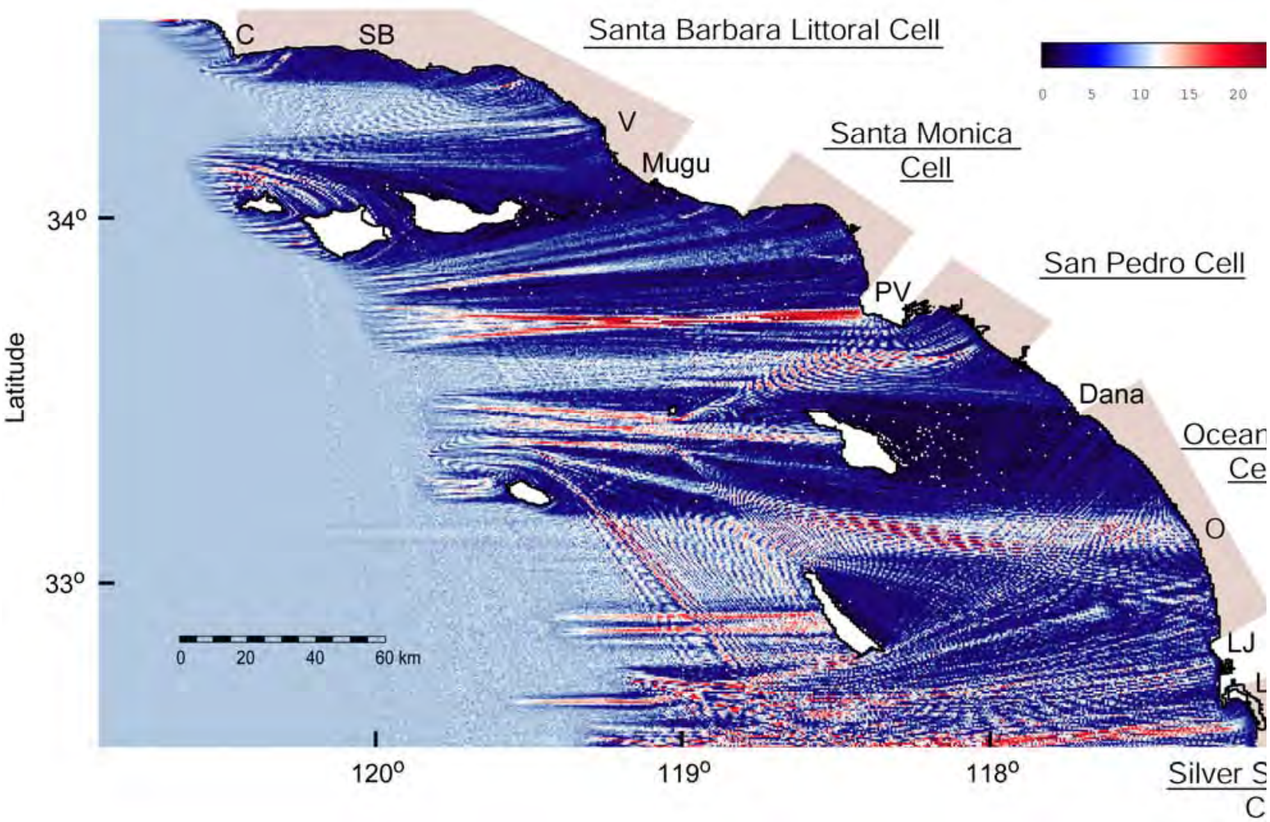


Figure 19. Back-refraction using *oceanrds.for* with waves measured by San Clemente CDIP station during the storm of 17 January 1988 with 10m high waves at 17 second period approaching the Southern California Bight from 270^0

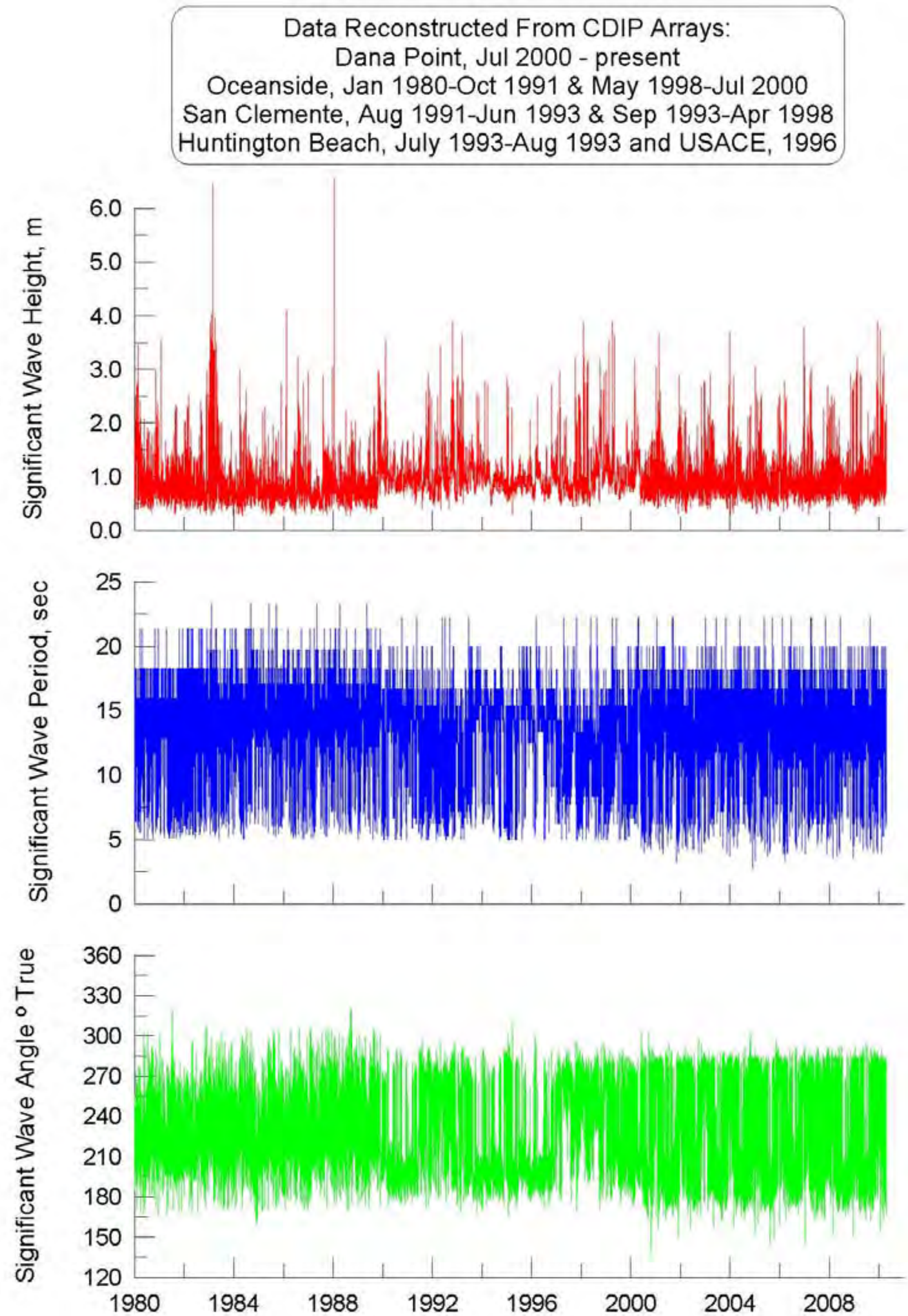


Figure 20. Deep water wave data for wave forcing in Torrey Pines Sub-Cell derived from back refraction of CDIP monitoring data, 1980-2012

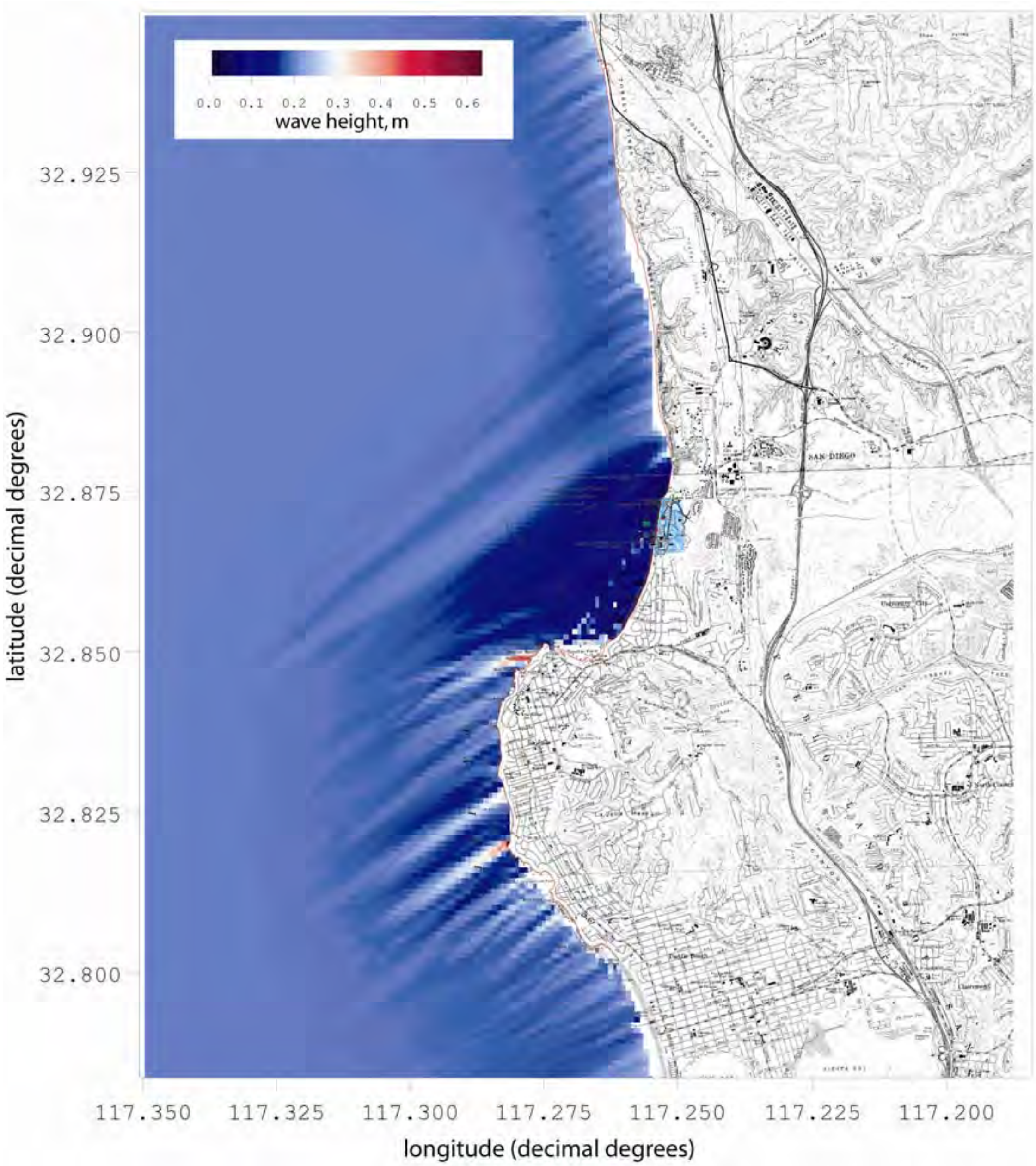


Figure 21. High resolution refraction/diffraction computation for extreme dry weather model scenario based on 0.2m deep water wave height from 210^0 with 10 sec period.

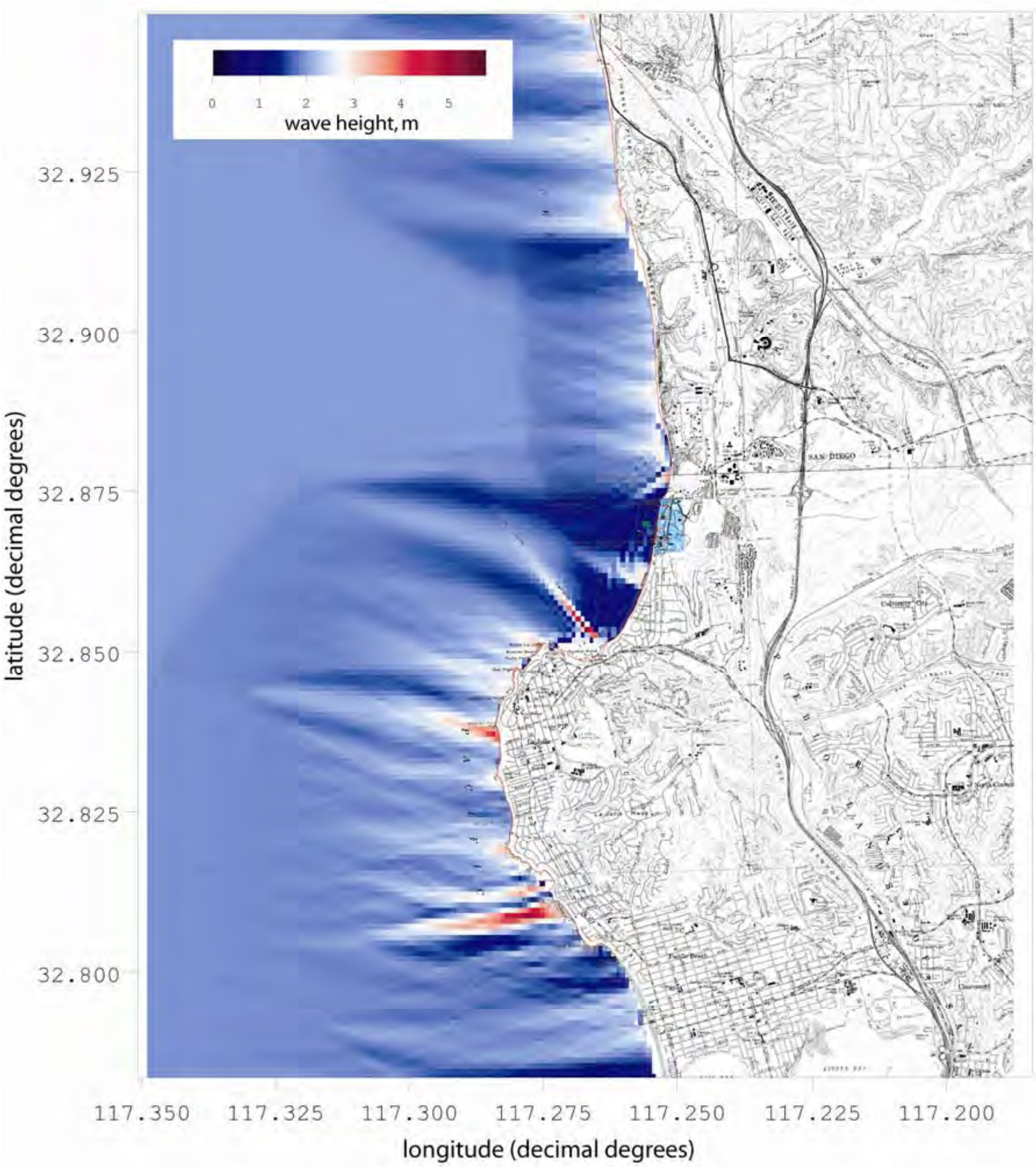


Figure 22. High resolution refraction/diffraction computation for worst-case wet weather model scenario based on 1.8 m deep water wave height from 283^0 with 14 sec period, 21 November 2011.

The directional dependence of the shoaling waves on La Jolla Shores that is shown in the refraction analysis of the examples in Figures 21 and 22 has been decomposed into a set of probability density functions in Figure 23, used to characterize the long-term wave mixing in the zone of initial dilution of storm drains SDL-062 and SDL-157. This figure reveals that the waves that influence dilution and dispersion of surf zone discharges of storm water at La Jolla Shores are ***bi-modal*** in character, meaning there are two primary directional modes: waves that approach the La Jolla Shores from the *northwest*, and waves that approach from the *southwest.* At lowest order, this bi-modality reflects seasonal wave climate cycles. The waves from the northwest are typically wet weather winter storm waves from Gulf of Alaska frontal-cyclones, and have typically higher significant wave heights 0(3m) and shorter wave periods 0(11 sec.). The southwest approaching waves are typically dry weather summer swells from distant Mexican tropical cyclones or southern hemisphere storms with lesser significant wave heights 0(1m - 2m) and longer wave periods 0(16 sec).

Southern Oscillation climate effects give rise to enhancements and protractions of the seasonal wave climate cycles, and their two extremes are referred to as El Niño (SOI negative) and La Niña (SOI positive). The wave climate in southern California changed, beginning with the El Niño years of 1978/79 and extending at least until the present. The average SOI for this period was -0.5, with the 1978/79 El Niño averaging -1.2, the 1982/83 El Niño averaging a record -1.7 and the 1993/94 El Niño recording a mean of -1.0. The prevailing northwesterly winter waves were replaced by high energy waves approaching from the west or southwest, and the previous southern hemisphere swell waves of summer have been replaced by shorter period tropical storm waves during late summer months from the more immediate waters off Central America. The net result appears to be a decrease in the southward component of the wave-induced longshore currents and the net littoral drift that had otherwise prevailed during the preceding thirty years (Jenkins and Wasyl, 2005, Inman and Jenkins, 1999). The wave statistics in Figure 23 seem to confirm this theory; whereby, despite the lesser intensity, there were 140,830 realizations of waves approaching from the southwest (El Niño dominant direction), as compared against only 61,508 realizations of waves approaching La Jolla Shores and the Torrey Pines Sub-cell from the northwest (La Niña dominant direction).

B) Currents: While waves dominate the initial dilution and dispersion of storm water and seawater discharges in the inshore domain, the tidal currents control dilution and dispersion in the offshore domain, particularly over most of the ASBS 29 footprint. A general southward net tidal drift is produced by the daily average of all the potential combinations of standing and progressive mixed tides. This net southward drift is an indication that the tidal transport in this region is *ebb dominated*. The strength of the net drift varies with the spring-neap cycle, with the strongest southward drift produced on the spring tides. In the La Jolla Bay portion of the Torrey Pines Sub-Cell, the southward drift is deflected by Pt La Jolla, producing a complex eddy structure in La Jolla Bay with a jet of converging flow immediately to the South of Pt La Jolla. The La Jolla Bay eddy is of particular interest because it often exerts sufficient entrainment near shore to cause ventilation of the ASBS by currents.

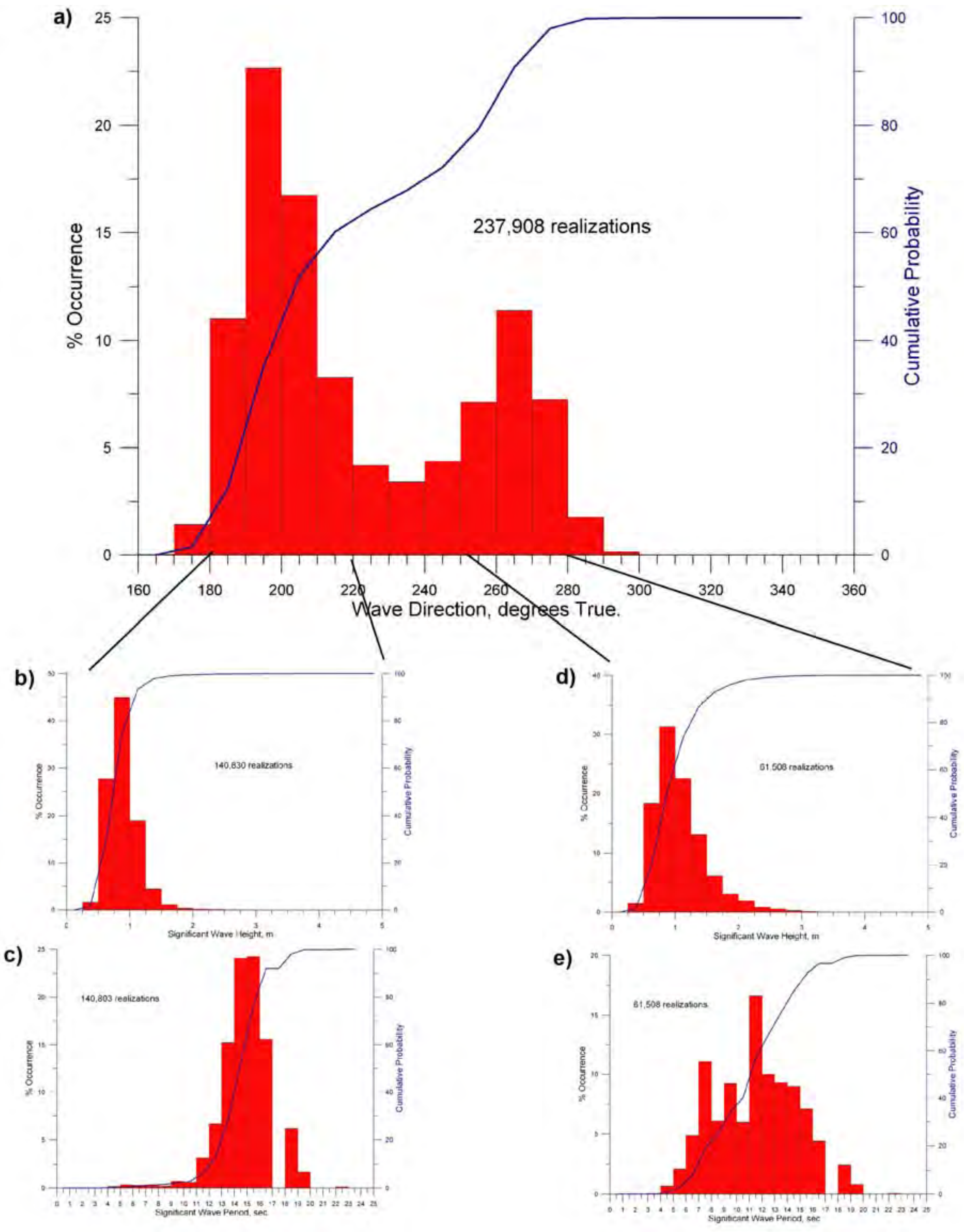


Figure 23. Statistics of composite wave record obtained by merging the CDIP archival data for 1980-2012 with the 2011-2012 ADCP wave burst measurements.

Figure 24 shows a progressive vector plot of the net tidal drift in La Jolla Bay during a neap tidal day, 22 August 2011. This current field is used to represent the minimal offshore mixing and advection conditions of the dry weather conditions. It can be seen from the vector field in Figure 24 that the La Jolla Bay eddy entrainment during neap tide does not extend close enough to shore to cause any appreciable ventilation of La Jolla Shores Beach or ASBS 29. The tidal drift ranges from nil to at most 5 cm/sec along the offshore boundary of the ASBS 29. The core of the La Jolla Bay eddy remains several kilometers offshore and its convergence with the broad field drift produces a 25 cm/sec jet flowing to the south near Marine Street in La Jolla. Given these features, the neap tidal drift used in the dry weather modeling scenario would appear to provide minimal dispersion of the beach discharges at La Jolla Shores Beach or the Devil's Slide area.

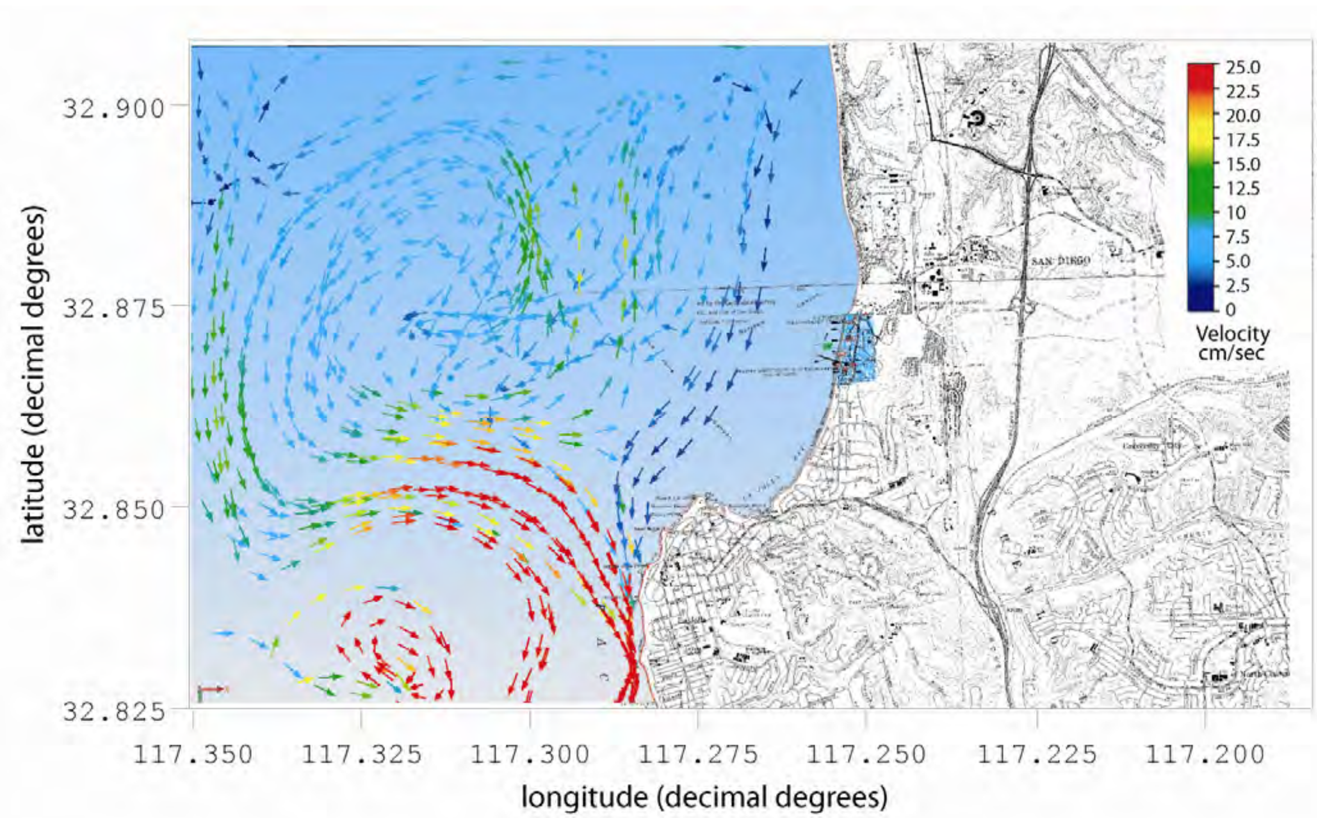


Figure 24. Progressive vector plot of current field in La Jolla Bay during a neap tidal day. Vector magnitude scaled by the color bar in the upper right corner.

By contrast, Figure 25 gives a progressive vector plot of the net tidal drift in La Jolla Bay during the spring tidal day concurrent with the wave conditions in Figure 22 during Wet Weather Event-1. This current field is used to represent the offshore mixing and advection conditions of the wet weather modeling scenario. In this case the La Jolla Bay eddy has moved closer to shore and has become paired with a system of counter rotating eddies off Pt La Jolla. The eddy entrainment currents off La Jolla Shores and through ASBS 29 are typically 9-15 cm/sec flowing toward the south, following the shoreline contours along La Jolla Shores and subsequently feeding an eddy pair off Pt La Jolla. This eddy pair in turn discharges a jet flowing 45 cm/sec into deeper waters several kilometers west of Pt La Jolla. In total, the eddy system in the La Jolla Bay region during spring tides forms a very efficient conveyor for transporting near shore discharges at La Jolla Shores Beach and Devil's Slide into deep water off shore with significant intervening vorticity and eddy mixing to promote dilution.

Figures 26-29 confirms these model results with ADCP current measurements taken at the northern edge of ASBS 29 (Figure 1) where local water depth is -10 m MSL. Figure 26 gives near bottom currents from profile cell #1 (2.4 m above seabed) during the site monitoring period 11/14/11-11/24/12 for the east-west current velocity component (a); north-south velocity component (b); total velocity amplitude (c). Figure 27 decomposes the near-bottom total velocity amplitudes into probability densities (red bars) and cumulative probability (blue). Over this one-year current monitoring effort, we occasionally find rather large maximum near bottom currents on the order of 50 cm/sec (~1.0 kt) at the northern

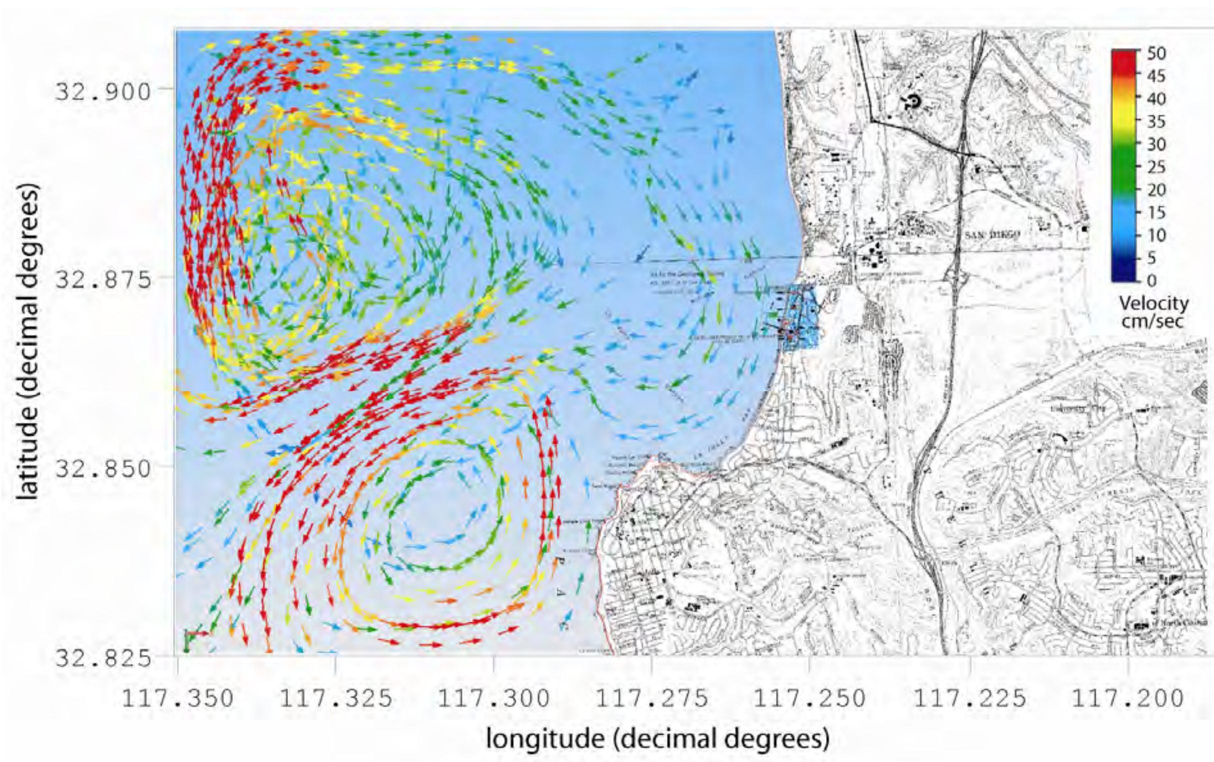


Figure 25. Progressive vector plot of current field in La Jolla Bay during a spring tidal day used in the wet weather modeling scenario. Vector magnitude scaled by the color bar in the upper right corner.

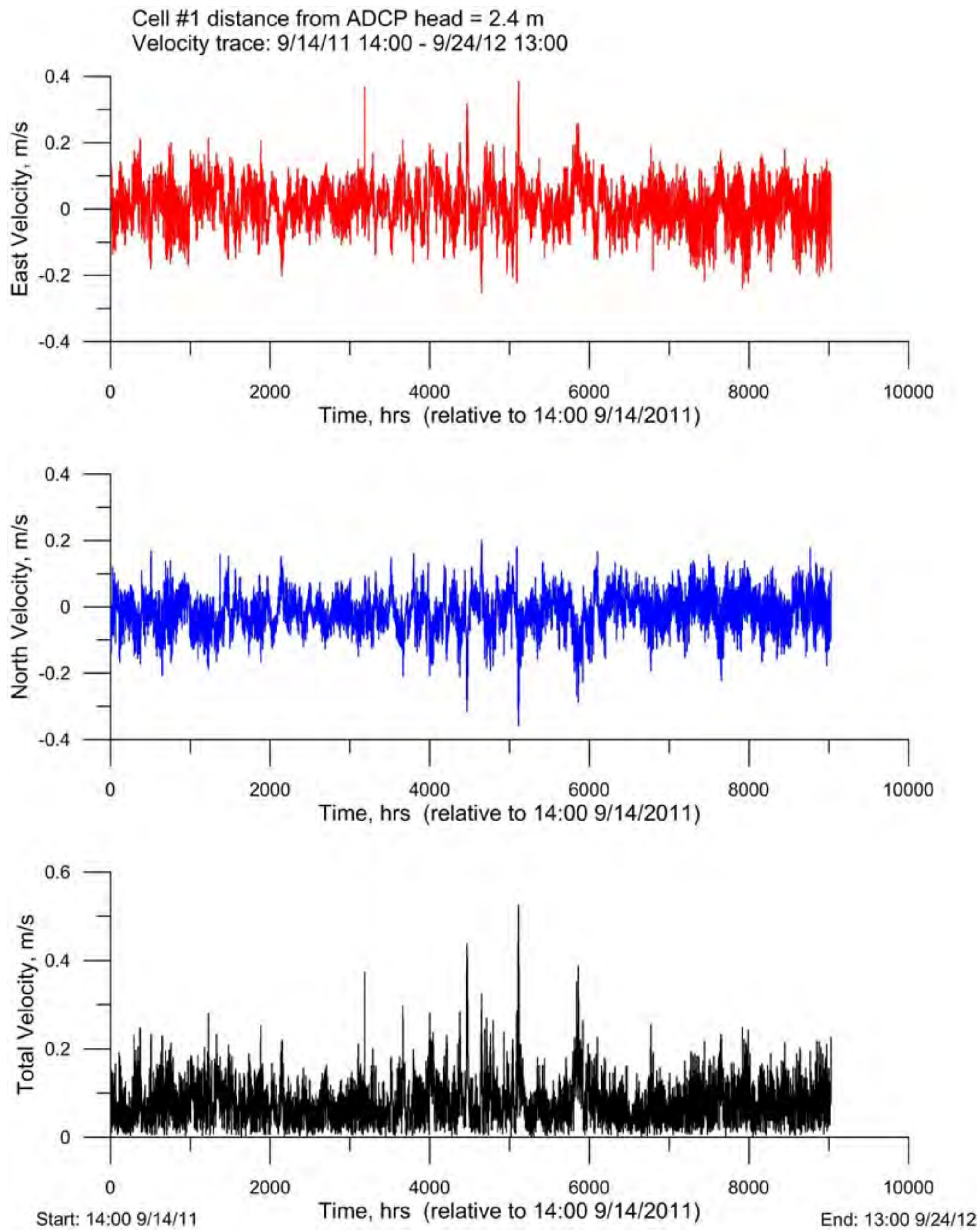


Figure 26. Near-bottom currents (2.4 m above seabed) at mooring location -10 m MSL at northern edge of ASBS 29 (cf. Figure 1). Measurements by Acoustic Doppler Current Profiler (ADCP), 11/14/11-11/24/12. East-west current velocity component (top); north-south velocity component (middle); total velocity amplitude (bottom).

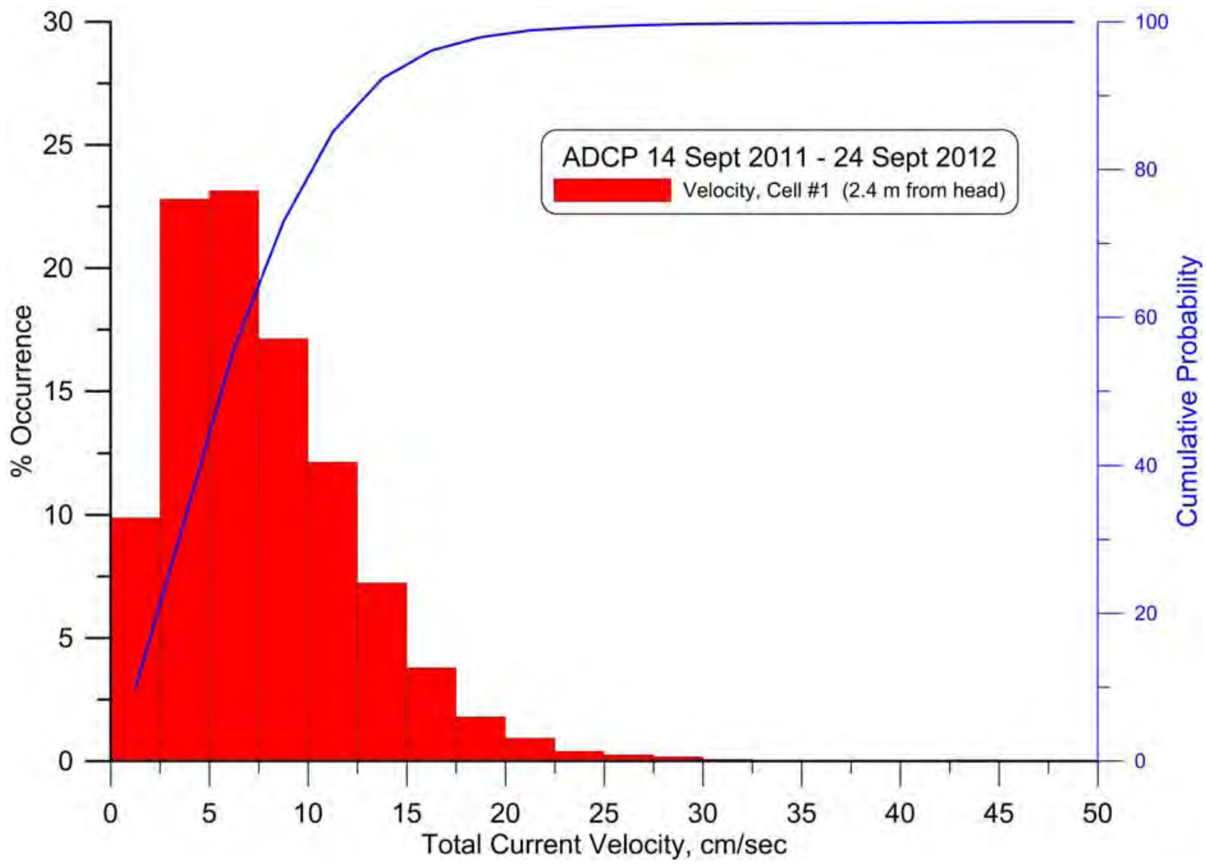


Figure 27. Histogram (probability density) and cumulative probability of near-bottom currents (2.4 m above seabed) at mooring location-10 m MSL at northern edge of ASBS 29, (cf. Figure 1); 11/14/11-11/24/12.

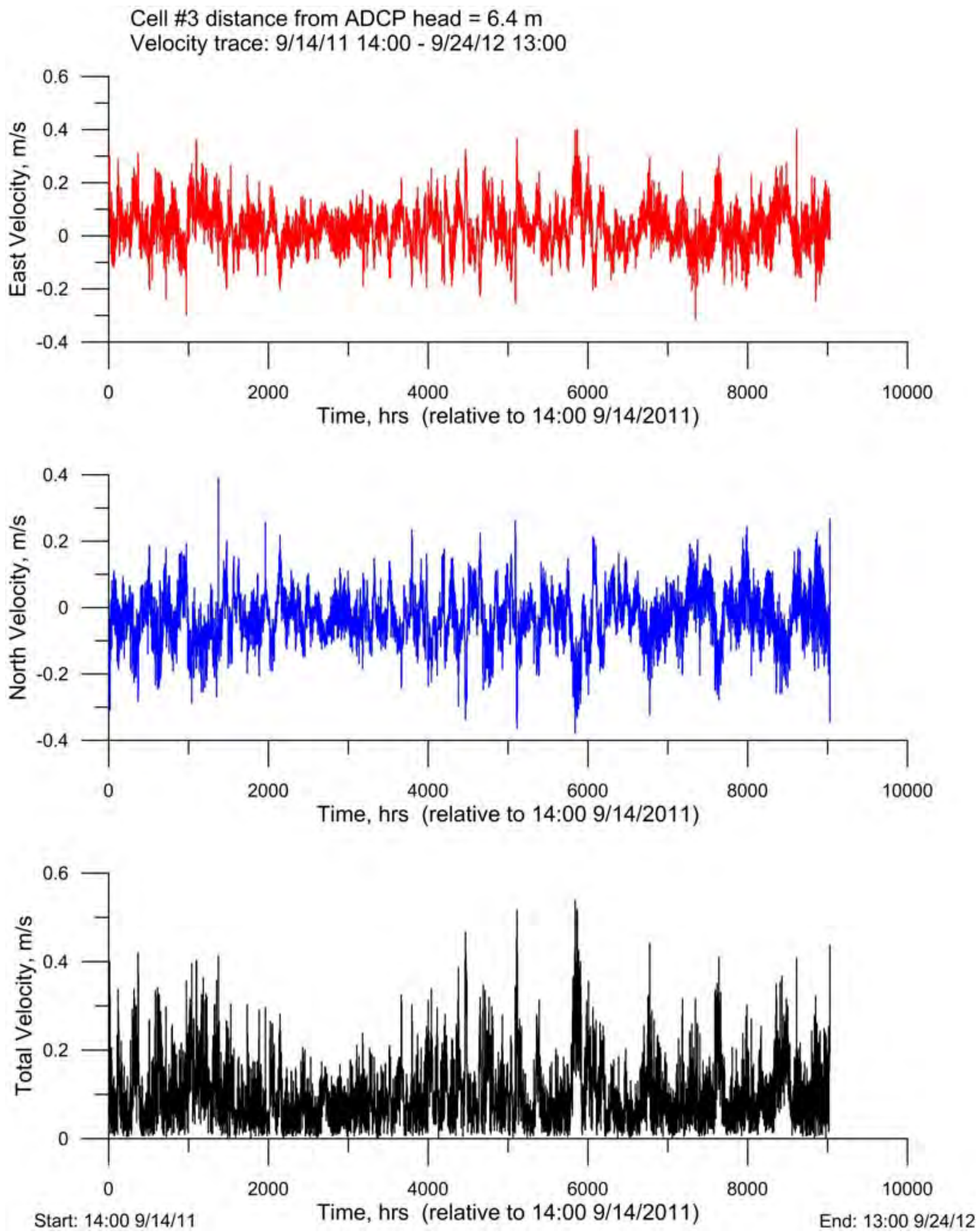


Figure 28. Interior currents (6.4 m above seabed) at mooring location -10 m MSL at northern edge of ASBS 29 (cf. Figure 1). Measurements by Acoustic Doppler Current Profiler (ADCP), 11/14/11-11/24/12. East-west current velocity component (top); north-south velocity component (middle); total velocity amplitude (bottom).

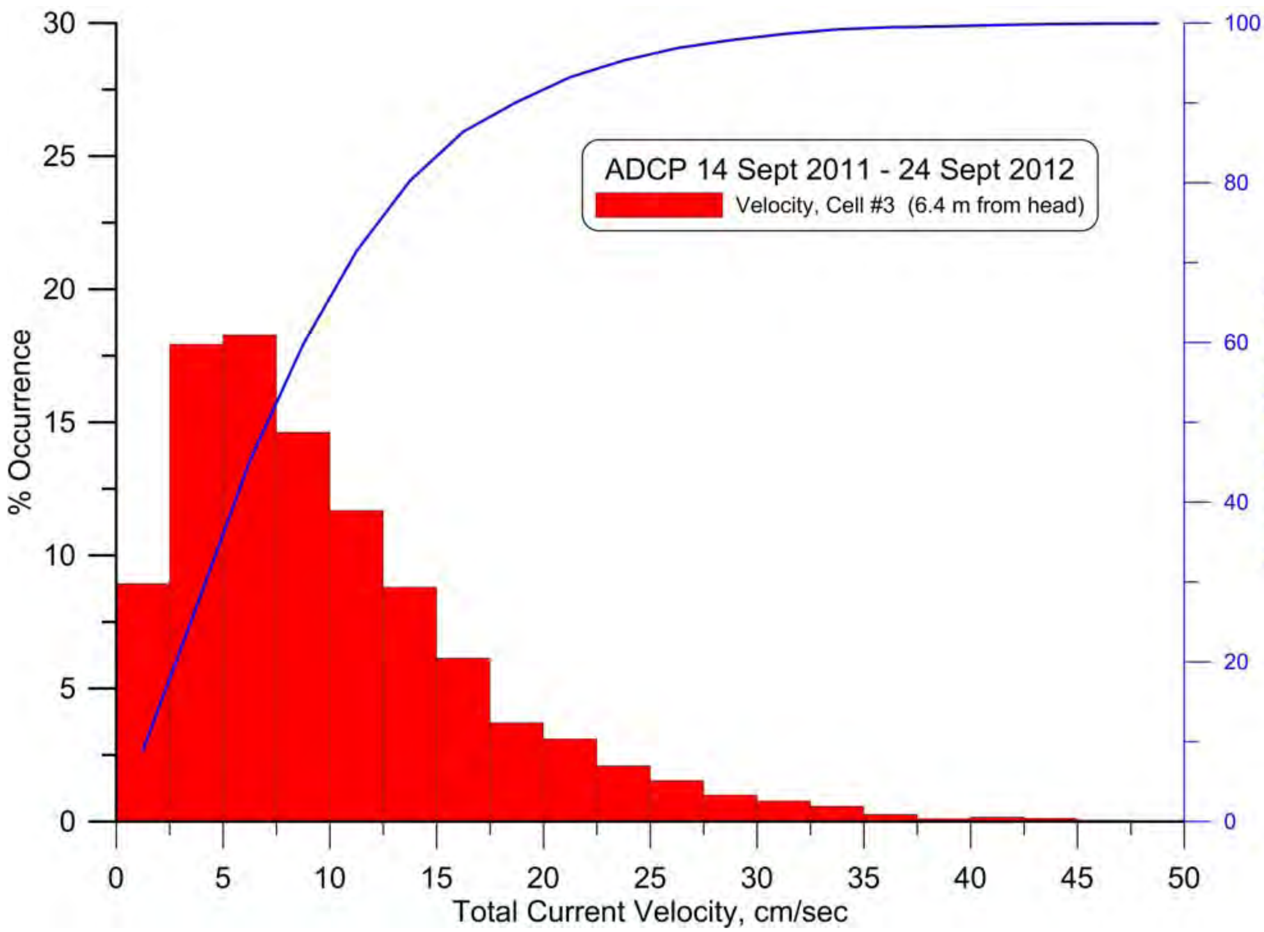


Figure 29. Histogram (probability density) and cumulative probability of interior currents (6.4 m above seabed) at mooring location-10 m MSL at northern edge of ASBS 29, (cf. Figure 1); 11/14/11-11/24/12.

edge of ASBS 29, while average near bottom currents are considerably less, on the order of only 5cm/sec. Further up into the water column, ADCP currents from profile cell #3 (6.4 m above seabed) in Figure 28 somewhat higher current amplitudes with more instances of maximum currents on the order of 50 cm/sec to 55 cm/sec during the site monitoring period 11/14/11-11/24/12. Accordingly, the probability densities (red bars) and cumulative probability (blue) of the interior currents in Figure 29 reveal higher average total current amplitudes, with average amplitudes of 8 cm/sec to 10 cm/sec, consistent with model results in Figures 24 & 25. These are favorable findings with respect to brine dilution rates of storm water in ASBS 29 where persistent currents are found at water column elevations close to the surface mixing layer of buoyant storm water.

C) Wind: Daily mean winds from the Scripps Pier Shore Station (SIO, 2001) and the Scripps Pier CDIP station (CDIP, 2002) were compiled over the 32 year period from January 1980 thru March 2012. This wind record is plotted in Panel-d of Figure 8. Because the lower Southern

California Bight is a "wind drought" region due to orographic blocking by the Peninsular Range, the 32 year mean wind speed is only 5.6 knots. However, El Niño storms and North Pacific cold fronts episodically increase daily wind speeds with the maximum sustained 24 hour mean wind speed reaching 19.6 knots during the 1997 El Niño storms. The minimum daily mean wind speed is 0 knots. In general the coastal winds in the nearshore of La Jolla Shores and Scripps Beach are benign, thereby limiting the degree of wind mixing of the surface water mass.

3.3 Model Scenarios

Based on discussions held in November 2004 with the San Diego Regional Water Quality Control Board, it was decided that the numerical modeling study of the beach discharges should be based on a wet weather worst-case scenario with a long-term probability assessment to bracket the envelope of the possible outcomes. The analysis of the multi-decadal rainfall variability of the San Diego found at the start if Section 3 provides justification for this approach.

A) Search Criteria for Wet Weather Worst Case Conditions: The 32 year long records of the boundary condition and variables in Figures 6-8 and the forcing function variables in Figure 20 were subjected to a joint probability analysis for the simultaneous recurrence of the combination of these variables for the ***wet weather worst case*** combination. Table 3 provides the search criteria that were applied to these records to establish the wet weather worst case combination of model inputs. The principle requirement of the wet weather worst-case is to maximize the combined discharges of storm water, while using ocean mixing variables that are characteristic, yet minimal for wet weather conditions. Because wet weather conditions occur during the passage of Pacific storms over the Southern California Bight, the worst case criteria of minimal ocean mixing tends to be mutually exclusive with the occurrence of storm water runoff. Wet weather worst case conditions for the receiving water involve storm minimums with respect to winds, waves and currents. To find such conditional minimums from among the 11,688 combinations found in Figures 6-8, and 20, the computer search criteria solved for the minimum of the largest 10% of the wind, waves and current combinations. Because the storm water discharge is predominately fresh water that is warmer than ocean water, the search criteria seeks salinity maximums and temperature minimums in the receiving water concurrent with the storm minimums of waves, winds and currents. In this way the maximum possible density contrast is achieved between the storm drain discharges and receiving water to retard the dilution rates of the discharge as much as possible under wet weather conditions.

Table 3.
Search Criteria for Wet Weather Worst-Case Combinations of Controlling Variables.

Variable	Search Criteria	Physical Significance
Discharge Rate	Maximize	Higher storm water and seawater discharge flow rates lead to higher source loading
Waves	Minimize Highest 10%	Smaller waves result in less mixing in surfzone and less inshore dilution
Currents	Minimize Highest 10%	Weaker currents result in less advection and less offshore dilution
Winds	Minimize Highest 10%	Weaker winds result in less surface mixing and less dilution in both the inshore and offshore
Ocean Water Level	Minimize	Lower water level results in less dilution volume in receiving water
Ocean Salinity	Maximize	Higher salinity leads to greater density contrast between discharge and receiving water
Ocean Temperature	Minimize	Higher temperature leads to density contrast between discharge and receiving water

B) Wet Weather Worst-Case Assignments: Among the three events monitored by AMEC (2012),Wet Weather Event-1 comes closest to matching the search conditions in Table-3 for the wet-weather worst-case proxy. The daily combination of receiving water variables that were recovered from the search criteria in Table 3 was represented best by the conditions on 21 November 2011. This day was post frontal with moderate, winds, waves, and currents. Ocean salinity was 33.11 ppt, depressed about 0.4 ppt by the run off from storms in the previous days while the ocean temperature was 14.5^{o} C, about 4^{o} C below the annual mean. Wave heights were 1.8 m, approaching La Jolla Bay from the northwest at 283^{0} with a 14 sec period. The refraction/diffraction pattern of these storm swells are shown in Figure 22. Winds were post-frontal northwest winds at 10-12 knots from at 290^{0}. The maximum tidal currents in ASBS 29 were 9-15 cm/sec flowing toward the south, (Figure 25), with a weak inshore counter current immediately outside the surf zone following the shoreline contours northward along La Jolla Shores. The tidal current was due to a moderate ebbing spring tide with a minimum water level of -3.63 ft NGVD. With this combination of receiving water variables the model overlaid the storm drain discharges from Wet Weather Event-1 from the monitoring program. Discharges for storm drain outfalls SDL-062 and SDL-157 were initialized according to Figures 9-11. For the purposes of initializing SDL-186 for the worst-case wet weather proxy, we use these modeled discharges from AMEC (2012). For Wet Weather Event-1, flow volume for SDL-186 was taken as 28,237 cubic feet. TSS fluxes for SDL-186 were taken as 70 kg in a 15 minute interval, or 4.7 kg/min, or equivalent to peak TSS fluxes from the neighboring SDL-062 storm drain.

C) Long-Term Simulations of Zone of Initial Dilution: The historic boundary conditions from Figures 6-8 and the forcing functions from Figure 20 were sequentially input into the model, producing daily solutions for the dilution field due to the peak flow event during the monitoring period (Wet Weather Event-1). This procedure inevitably produces some combinations of wet weather discharges with dry weather ocean mixing conditions, and thereby results in certain worst case dilution scenarios that are more extreme, with even less dilution, than the wet weather worst case proxy derived from the monitoring program (Wet-Weather Event-1). The input stream of seven controlling variables from Figures 6-8 & 20 produced 11,688 daily solutions for the dilution field in the zone of initial dilution (ZID) taken as the surfzone. A numerical scan of each of these daily solutions searched for the minimum dilution averaged between the wave break point and the shoreline along the inshore boundary of ASBS 29. Based on the ZID definition contained in the NPDES permit issued by the RWQCB, San Diego Region, for beach discharges at Encina Generating Station, Carlsbad, CA, the search for the minimum dilution was taken longshore to a distance of 1000 ft (305 meters) away from the discharge points in all directions, (although the minimum dilution was found without exception to be within 150m of the discharge points). The solution scans searched for minimum dilution in both the water column and along the sea floor. For each search, the minimum dilution found in any direction away from the outfall was entered into a histogram bin for ultimately assembling a probability density function and cumulative probability from the 11,688 outcomes. These results provided a statistical basis for assessing long term variability of dilution in the ZID. For comparison, a similar procedure of searching the solution space for minimum dilution was repeated at an offshore control point near the interior of ASBS 29, where local water depth was -10m MSL, see Figure-1.

3.4 Calibration

The coupled sets of wave, current and dilution/dispersion models were calibrated for end-to-end simulations of known dye dispersion events measured off Scripps Beach by Inman et al. (1971). Initializations for the model were derived from the measured forcing functions reported in that publication. Free parameters in the subroutines of the **SEDXPORT** for dilution/dispersion model were adjusted iteratively until a best fit was achieved between the measured and simulated dye concentrations and dilution factors.

The subroutines of **SEDXPORT**.for contain seven free parameters which are selected by a calibration data set specific to the coastal type for which the hindcast calibration simulation is run. These parameters are as follows according to subroutine:

BOTXPORTAf

- *ak2 - stretching factor for vertical eddy diffusivit
- *ak - adjusts mixing lengths for outfalls

NULLPOINTAf

*ak7 - adjusts the asymmetry of the bedform distribution curve

SURXPORTAf

*aks - adjusts the surf zone suspended load efficiency, K_s

ak4 - stretching factor for the horizontal eddy $diffusivity_x$

RIVXPORTAf

*ak3_1 - adjusts the jetty mixing length and outfall mixing lengths

*ak3 - stretching factor for the horizontal eddy diffusivity of the river $plume_H$

The set of calibration values for these parameters was used without variation or modification for all model scenarios.

4.0 RESULTS

The model scenarios defined in Section 3.3 are run in simulations of one tidal day using the numerical codes described in Section 2 and listed in Appendices A-F. The dilution fields are then depth averaged, and averaged over the simulation period. In the sections that follow, dilution fields are contoured in base -10 log according to the color bar scale in the lower right-hand corner of each plot, with a scale range that covers dilution factors between 10^0 and 10^7. Two separate perspectives of the worst-case results are given: 1) a composite dilution field when storm drains SDL-062, SDL-157 and SDL-186 are discharging simultaneously (Figure 30); and 2) the dilution field associated with each individual storm drain discharging in isolation under worst-case conditions (Figures 31-33). These dilution results are then applied to the peak TSS discharge fluxes to give TSS concentrations in the receiving water in mg/L in Figures 34-36.

4.1 Wet Weather Worst Case Dilution Results

Figure 30 shows the broad-scale view of the footprint of the dilution field for the wet weather worst case scenario (Wet-Weather Event-1) involving the simultaneous discharge from storm drains SDL-157, SDL-062 and SDL-186. In the broad-scale view, the dilution field of storm water spreads about a kilometer seaward of the shoreline due to vigorous cross shore mixing and advection from the storm winds and shoaling swells of the wet weather receiving water scenario. The dilution field of storm water from all three storm drains spreads about 2 kilometers along the shoreline, covering most of ASBS 29 and intruding into ASBS 31 under the advective influence of the longshore transport of rip current cells near the shoreline, while the tidal drift dominates the longshore spreading of the dilution field offshore, (cf. Figures 22 and 25). The dilution factors for the combined discharges from SDL-157, SDL-062 and SDL-186 in both ASBS 29 and ASBS 31 range between a minimum of 10^2 near shore to 10^7 along the seaward boundaries during the wet weather worst case. Dilution factors of 10^4 to 10^7 characterize the outer one-half of ASBS 29, while dilution factors 10^2 to 10^4 characterize the inner one-half. In the immediate neighborhood of SDL-157 and SDL-062, very near the shoreline of ASBS 29, dilution factors are as low as 40 to one.

Figures 31-33 allow us to examine the individual contributions to the combined storm water plume from each storm drain during the wet weather worst case from the monitoring program. It is clear from Figures 31 and 32 that the preponderance of the discharge plume is from SDL-157 and SDL-062, while SDL-186 at Devil's Slide (Figure 33) makes a very minor contribution. This is a fortuitous outcome in the sense that a significant amount of high-value, hard bottom marine habitat lives in the southern end of ASBS 29, and in the immediate neighborhood of SDL-186. Dilution factors of storm water discharged from SDL-186 are at most 10^3 near the shore in the nearfield of SDL-186 (Figure 33) and more typically 10^4 to 10^5 elsewhere along the bluff-faced shoreline of the southern end of ASBS 29. The largest single contributing footprint to the combined storm water plume appears to be from SDL-062 at the end of Avendia De La Playa (Figure 32) that produces a large patch with 10^2 minimum dilution very near the shoreline that spreads south along the shore to the La Jolla Beach and Tennis Club. The region influenced by

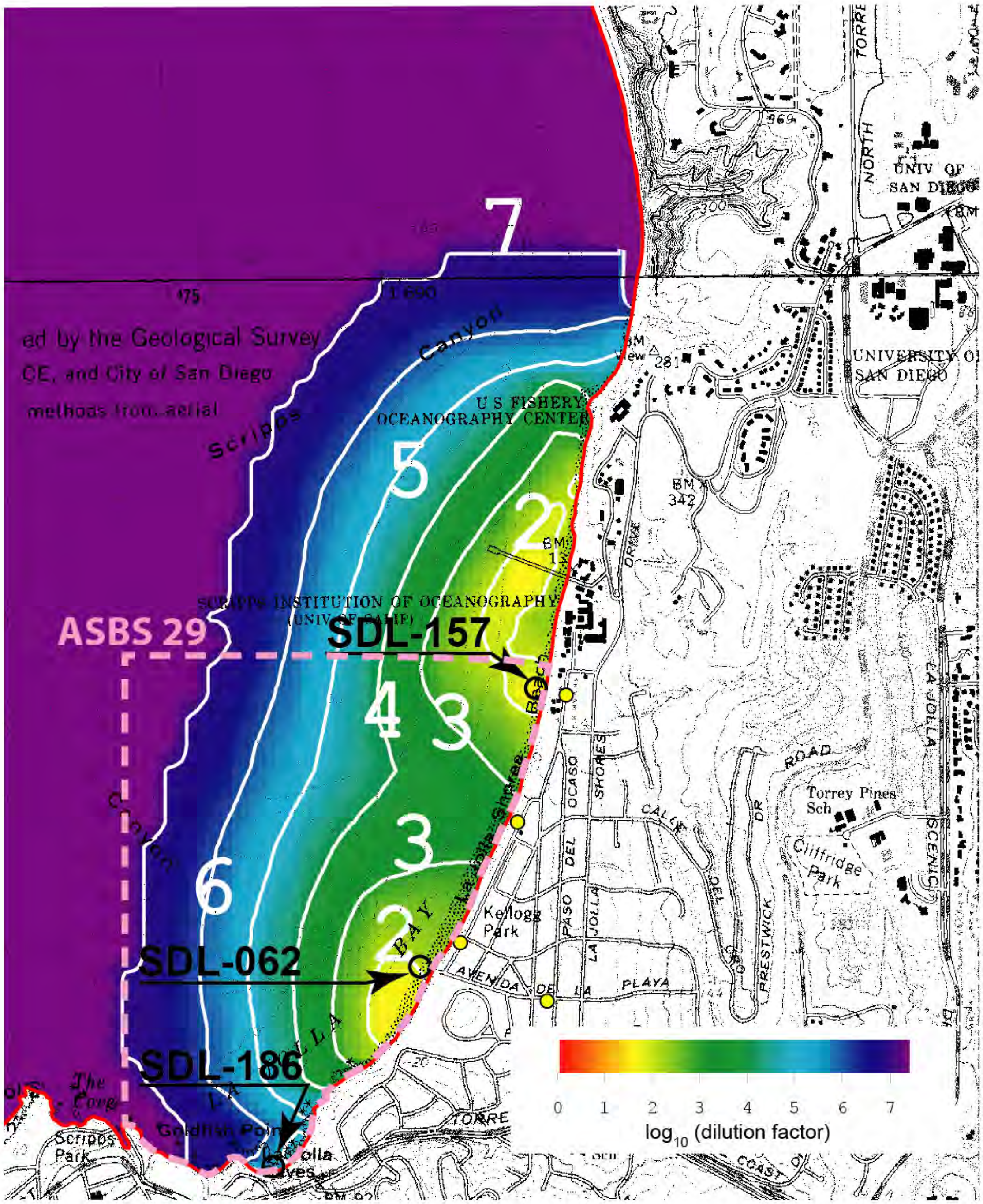


Figure 30. Depth averaged dilution for wet weather worst case due to simultaneous discharges of storm water from SDL-157, SDL-062 and SDL-186; 21 November 2011; waves: H = 1.8 m, T = 14 s, α = 283^{o}; wind = 11 kts.

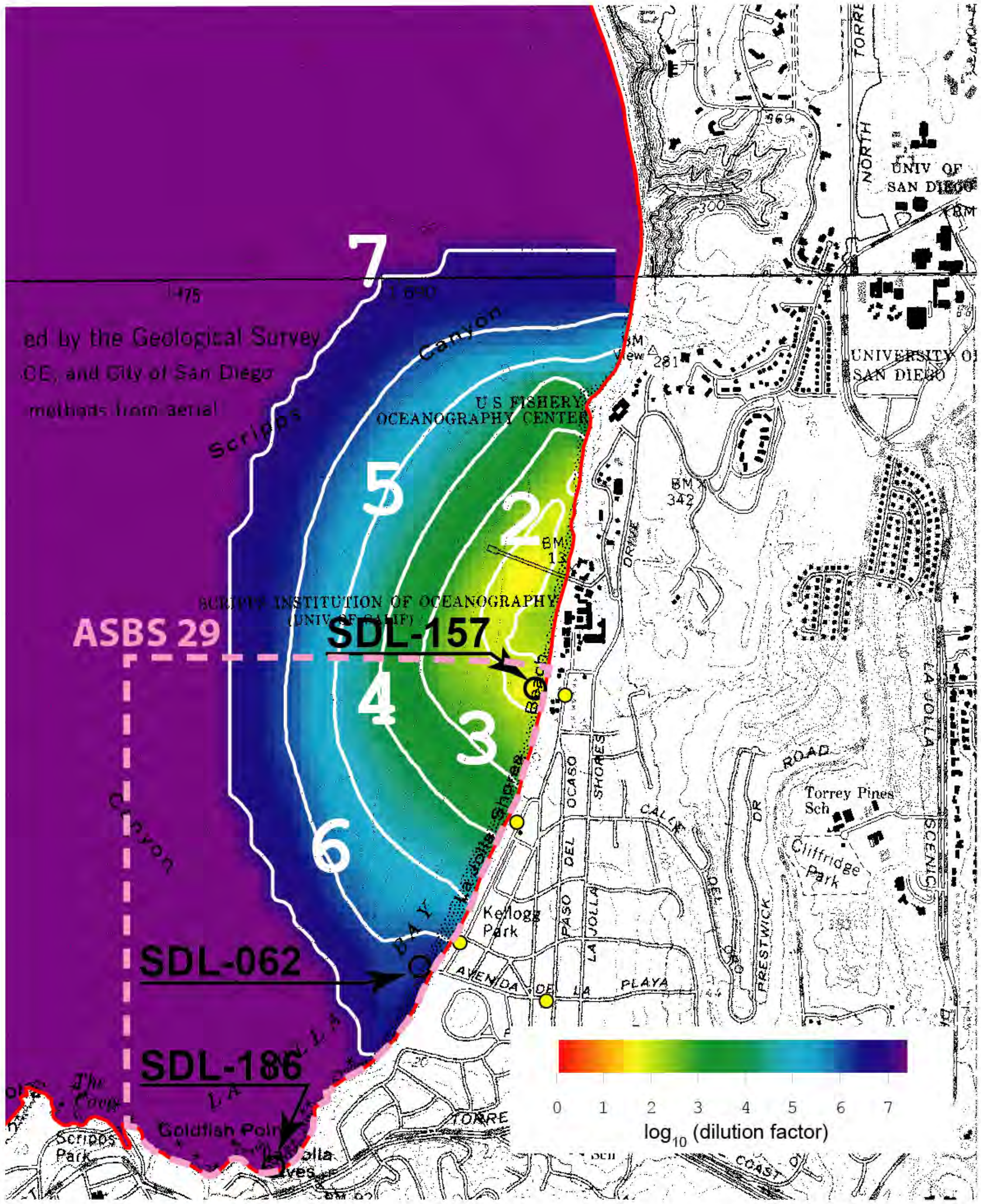


Figure 31. Depth averaged dilution for wet weather worst case due to isolated discharge of storm water from SDL-157; 12 November 2011; waves: H = 1.8 m, T = 14 s, α = 283°; wind = 11 kts.

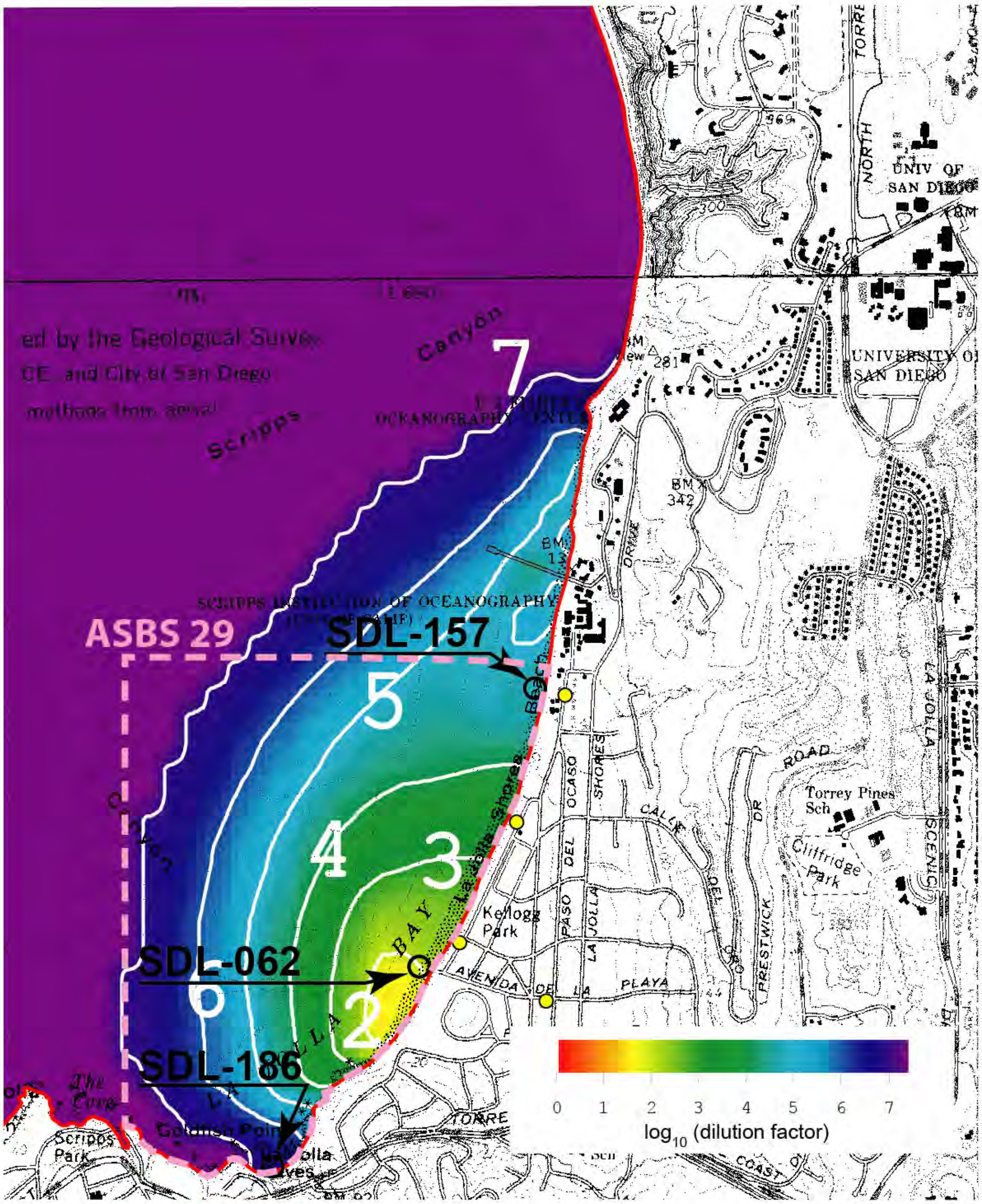


Figure 32. Depth averaged dilution for wet weather worst case due to isolated discharge of storm water from SDL-062; 21 November 2011; waves: H = 1.8 m, T = 14 s, α = 283^{o}; wind = 11 kts.

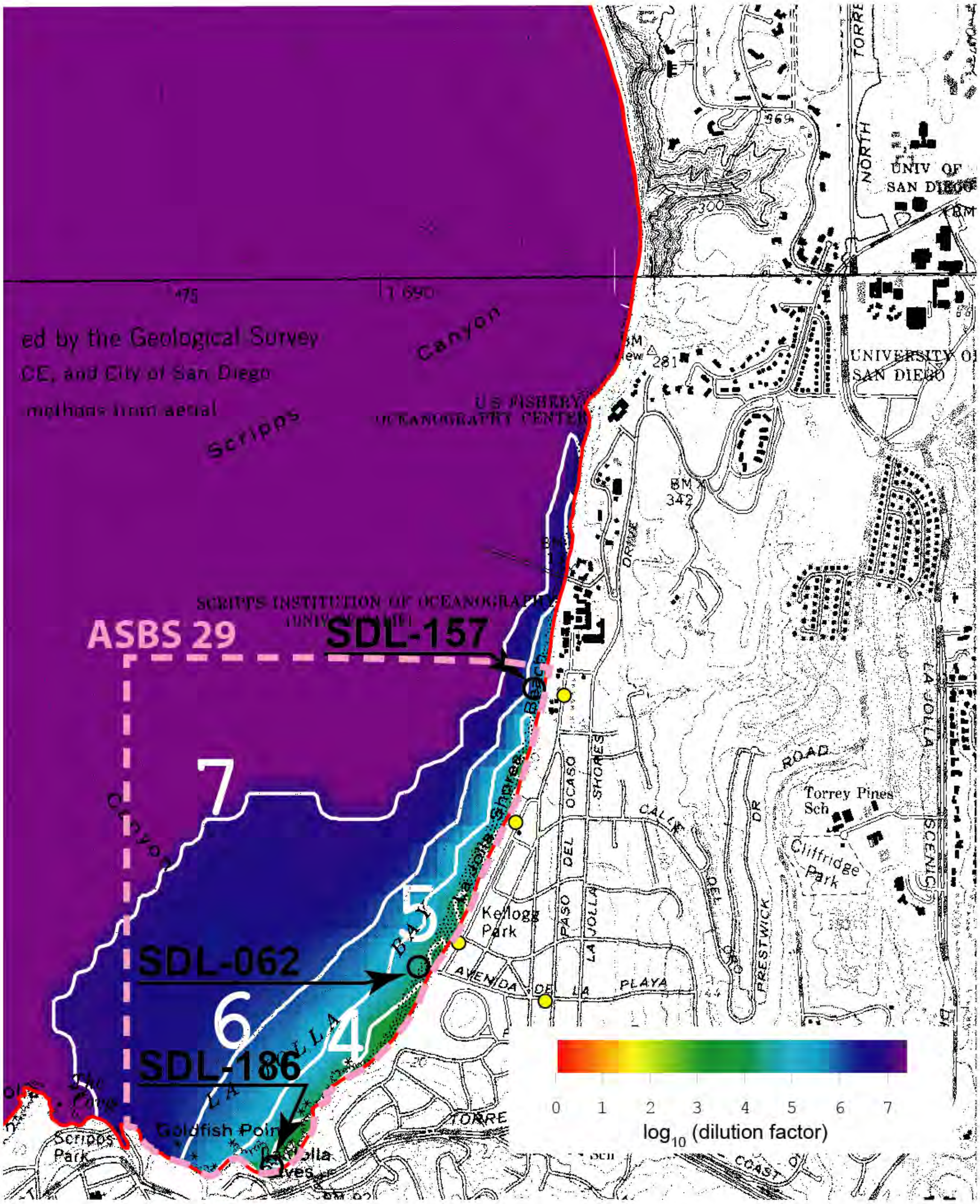


Figure 33. Depth averaged dilution for wet weather worst case due to isolated discharge of storm water from SDL-186; 21 November 2011; waves: H = 1.8 m, T = 14 s, $\alpha = 283^{o}$; wind = 11 kts.

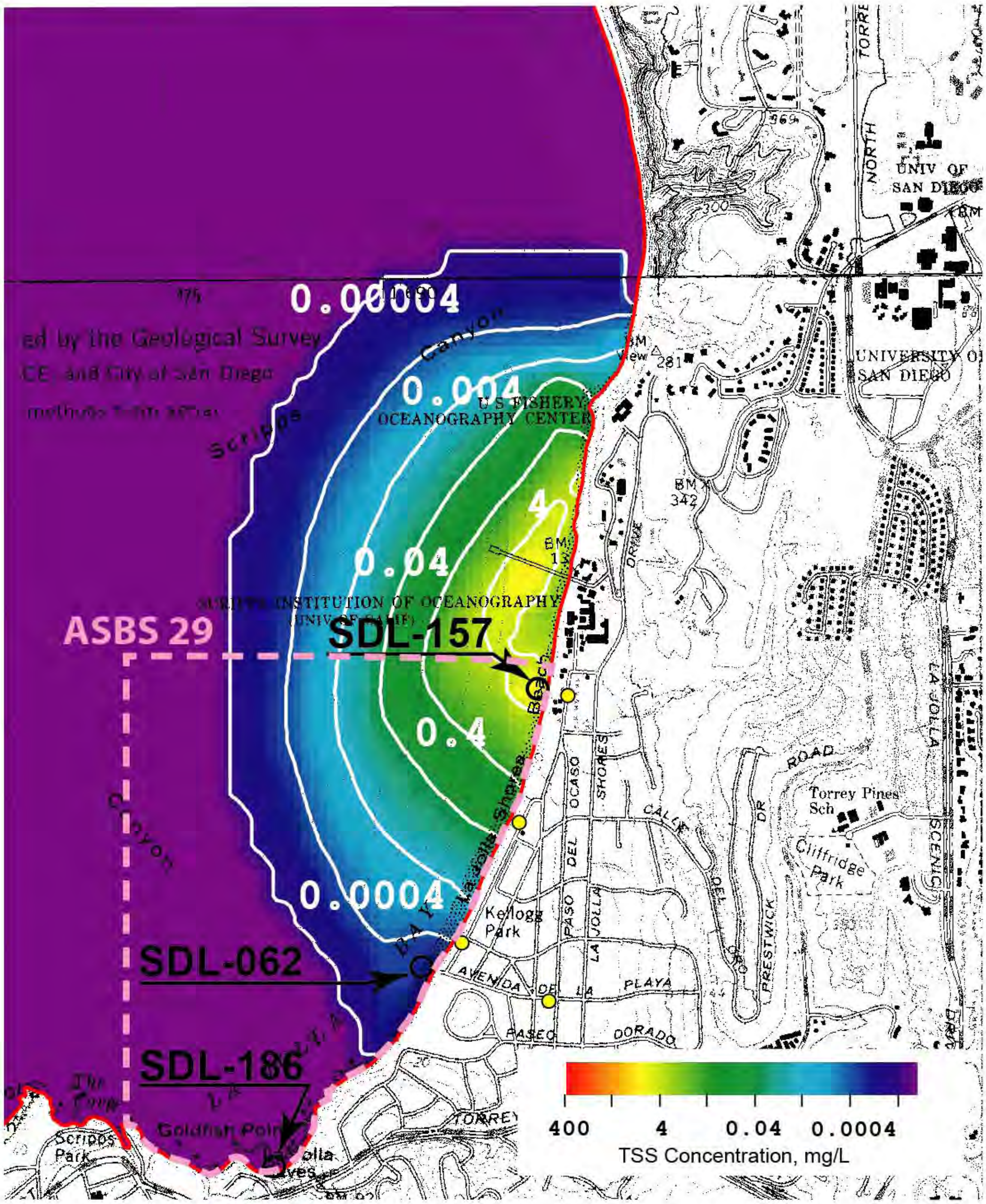


Figure 34. Depth averaged concentration of total suspended solids (TSS) for wet weather worst case due to discharge of storm water from SDL-157; 21 November 2011; waves: H = 1.8 m, T = 14 s, α = 283°; wind = 11 kts.

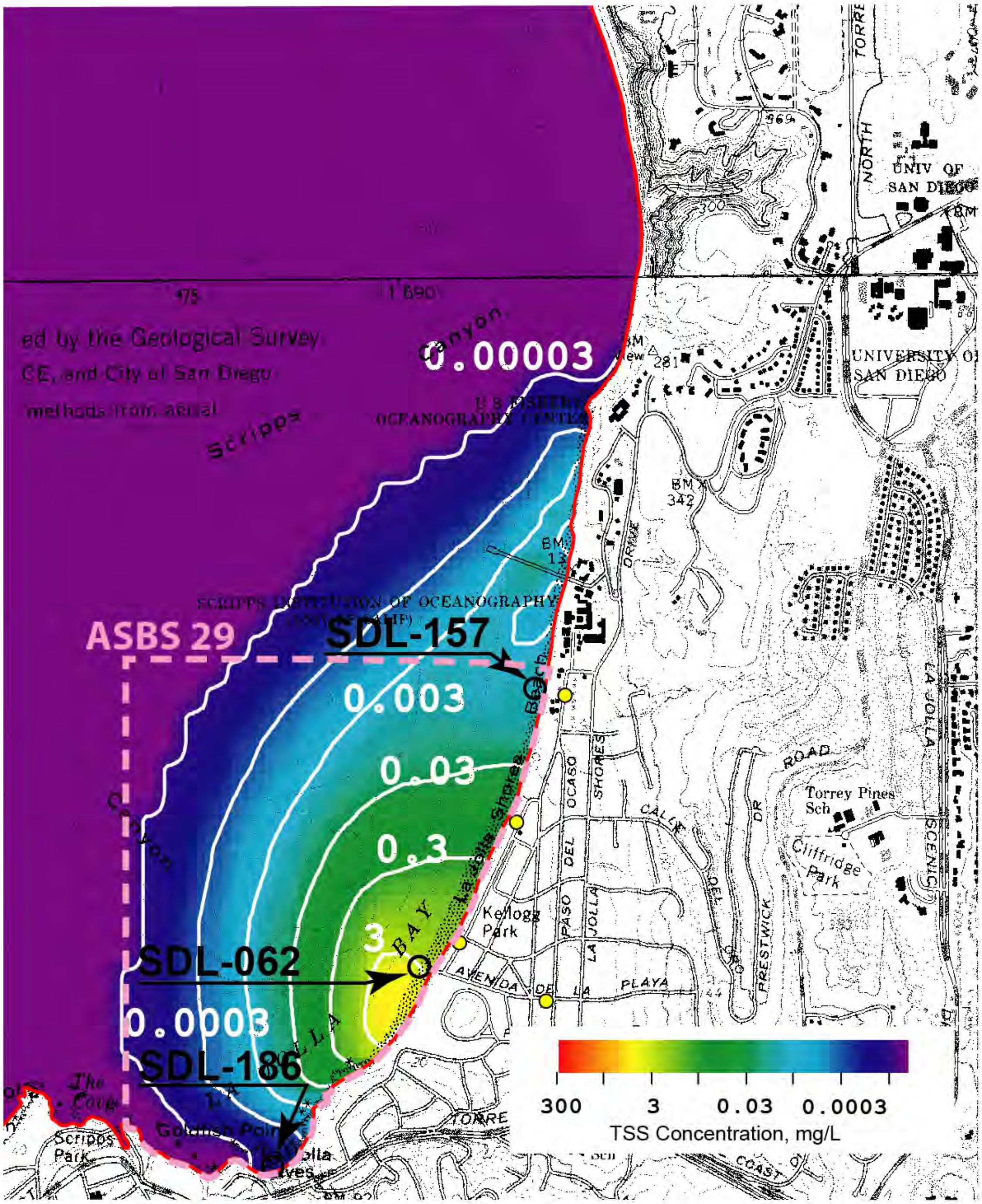


Figure 35. Depth averaged concentration of total suspended solids (TSS) for wet weather worst case due to discharge of storm water from SDL-062; 21 November 2011; waves: H = 1.8 m, T = 14 s, α = 283°; wind = 11 kts.

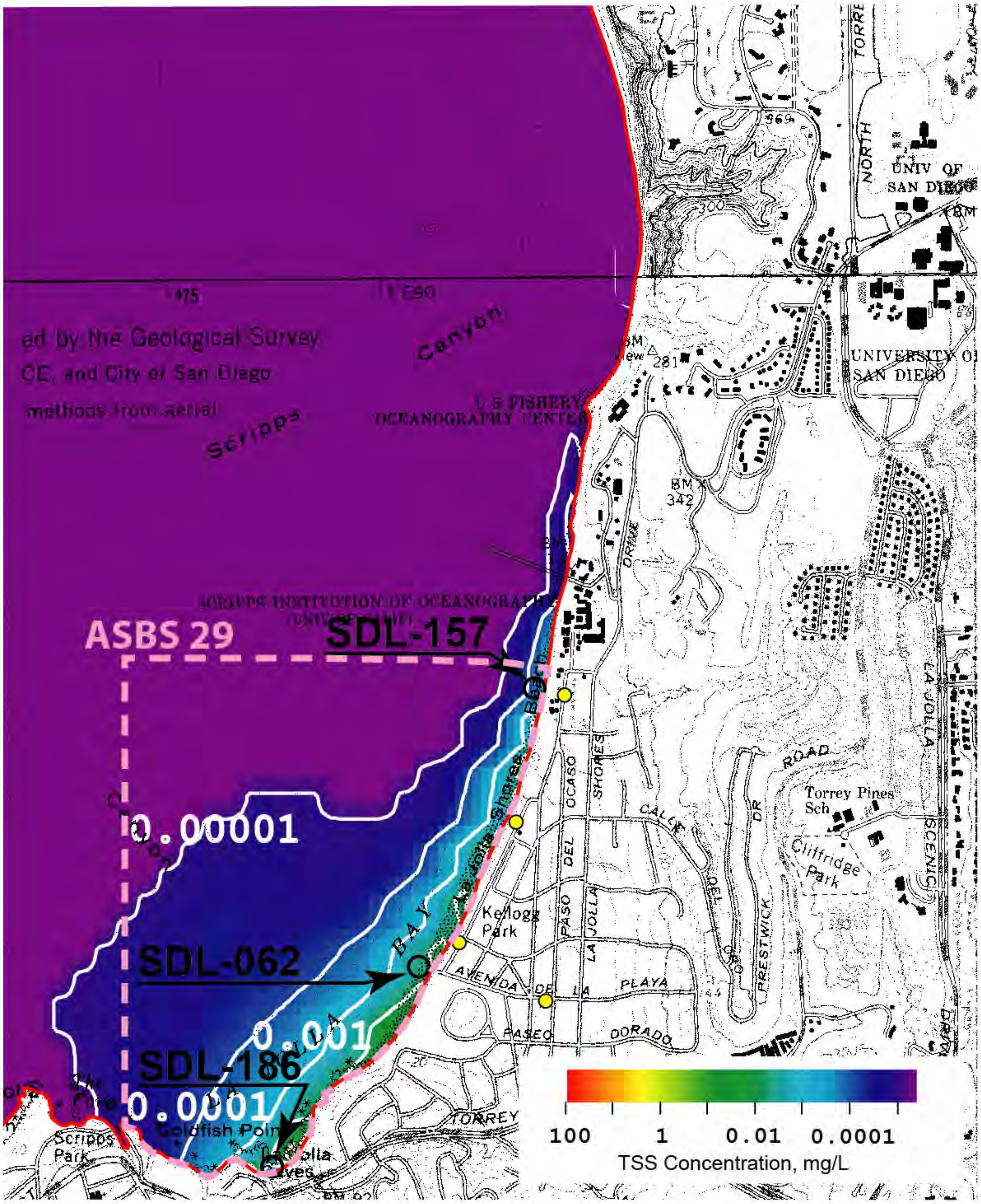


Figure 36. Depth averaged concentration of total suspended solids (TSS) for wet weather worst case due to discharge of storm water from SDL-186; 21 November 2011; waves: H = 1.8 m, T = 14 s, α = 283°; wind = 11 kts.

this feature from SDL-062 is sandy, soft-bottom marine habitat. SDL-157 produces another patch with 10^2 minimum dilution, but this feature moves offshore and intrudes into the southern end of ASBS 31 (Figure-31), where again the marine habitat is a sandy soft-bottom type.

Figures 34-36 the corresponding concentrations of total suspended solids (TSS) in the storm water plumes discharged by the individual storm drains during the wet weather worst case from the monitoring program. Maximum TSS concentrations in the receiving water due to SDL-186 storm water discharges are at most 0.001 mg/L near the shore in the nearfield of SDL-186 (Figure 36) and more typically 0.0001 mg/L elsewhere along the bluff-faced shoreline of the southern end of ASBS 29, where significant high-value, hard bottom marine habitat resides. The highest TSS concentrations in the receiving water around SDL-062 (at the end of Avendia De La Playa, see Figure 35), were found to be 3 mg/L in a nearshore patch that that spreads south along the shore to the La Jolla Beach and Tennis Club. Most of the shoreline impacted by SDL-157 experiences maximum TSS concentrations from storm water in the range of 0.3 mg/L to 0.03 mg/L, all of which is sandy soft- bottom marine habitat. At the northern end of ASBS 29, and extending into ASBS 31, maximum TSS concentrations in the receiving from SDL-157 are in the range of 4 mg/L to 0.4 mg/L, the highest anywhere in La Jolla Bay. These relatively higher TSS values are probably a consequence of the steep land forms that comprise the watershed of SDL-157, with portions of both developed and undeveloped coastal bluffs. Regardless, the marine habitat subjected to these TSS loadings from SDL-157 is sandy, soft-bottom habitat type.

4.2 Long Term Minimum Dilution in the Surf Zone and Offshore

Because of the disparity in length scales between dilution in the offshore region versus the surfzone, a separate analysis was performed on a nested fine scale grid covering the surf zone for a long shore reach that extended 1000 ft (305 meters) either side of the beach outfalls. The size of this grid was based on precedent already set for the definition of a zone of initial dilution (ZID) by the Regional Water Quality Control Board, San Diego. All 11,688 combinations of receiving water variables from Figures 6 and 20 were input for daily simulations of dilution using the numerical codes described in Section 2 and listed in Appendices A-F. The ZID was searched for the minimum dilution after averaging across the width of the surf zone. Source loading for these simulations were based on the discharges measured during Wet-Weather Event-1 from outfalls SDl-157, SDL-062 and SDL-186. The surfzone dilution results are then compared to minimum dilution on the sea surface at an offshore control point in ASBS 29 where local water depth is -10 m MSL (Figure-1).

Figure 37 presents a probability density function (red histogram) with corresponding cumulative probability (blue line) of the minimum surfzone dilution within the ZID for storm drain SDL-157 for an ensemble of 11,688 daily outcomes (32 years). The median outcome is a minimum dilution factor of 32 to 1 based on the historical sequence of receiving water variables found in Figures 6-8 and 20. However the potential range of minimum dilution within the ZID goes as high as 88 to 1, and as low as 15 to 1. Low energy, dry weather ocean mixing produced the 15 to 1 minimum dilution outcome, which had a probability of occurrence of 0.13%. Altogether, 90% of the potential outcomes produce minimum dilutions in the ZID greater than 20 to 1.

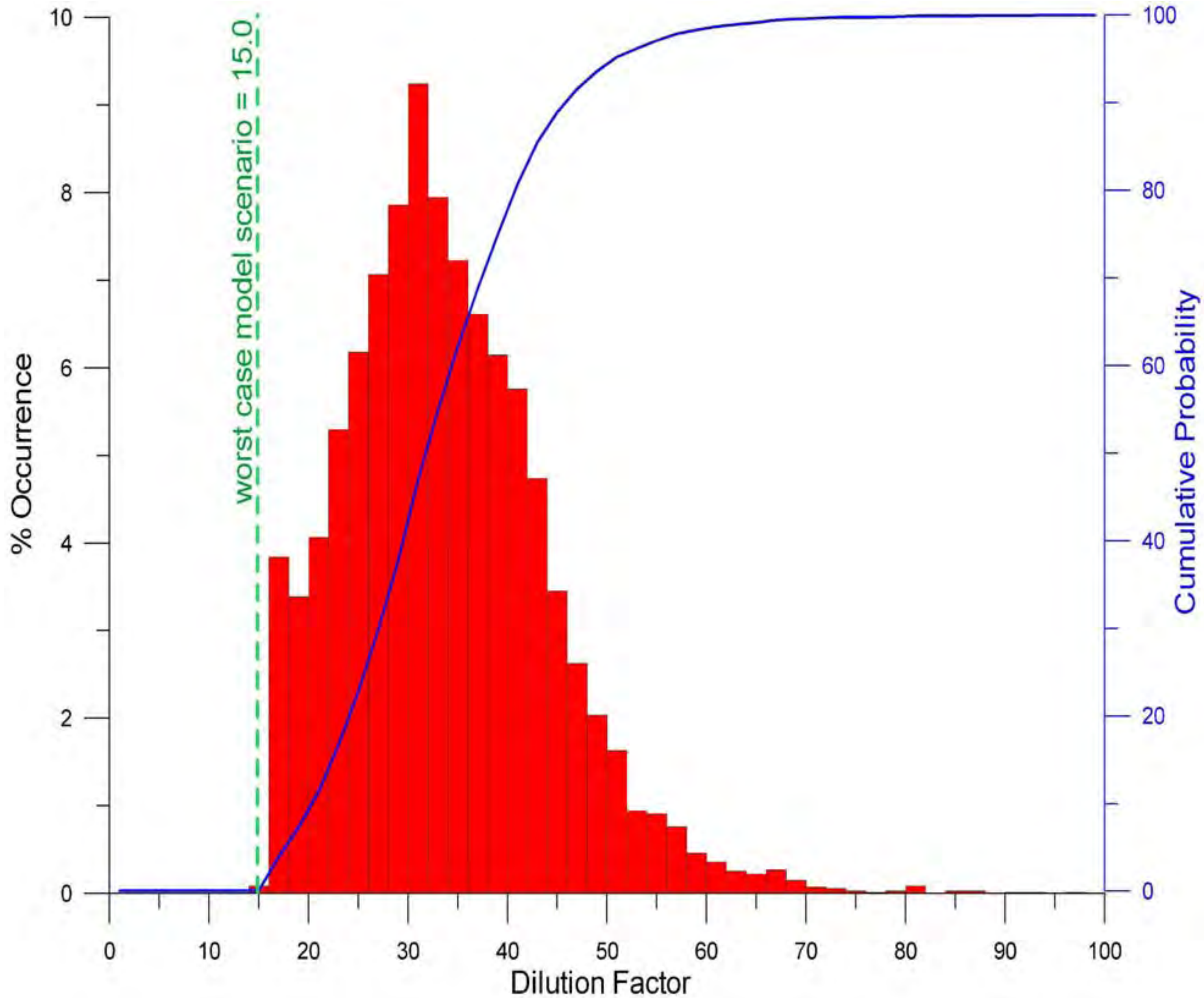


Figure 37. Probability density function (red) with corresponding cumulative probability (blue line) of the minimum surfzone dilution within the ZID around storm drain SDL-157. Derived from ocean boundary conditions and forcing functions, 1980-2012, 11,688 total numbers of realizations.

Minimum surfzone dilutions in the ZID off SDL-062 in Figure 38 are slightly less. This is due to the wave shadow found chronically in the refraction pattern of La Jolla Shores in the neighborhood of Avendia De La Playa (Figures 21 & 22). The minimum surfzone dilution of discharges from SDL-062 is found in Figure 38 to be 12.6 to 1, again a low energy, dry weather ocean mixing result. The median outcome of minimum surfzone dilution for SDL-062 is 22 to 1; while the potential range of minimum dilution goes as high as 64 to 1, based on the historical sequence of receiving water variables found in Figures 6-8 and 20. Ninety percent of the potential outcomes for SDL-062 produce minimum surfzone dilutions greater than 15 to 1.

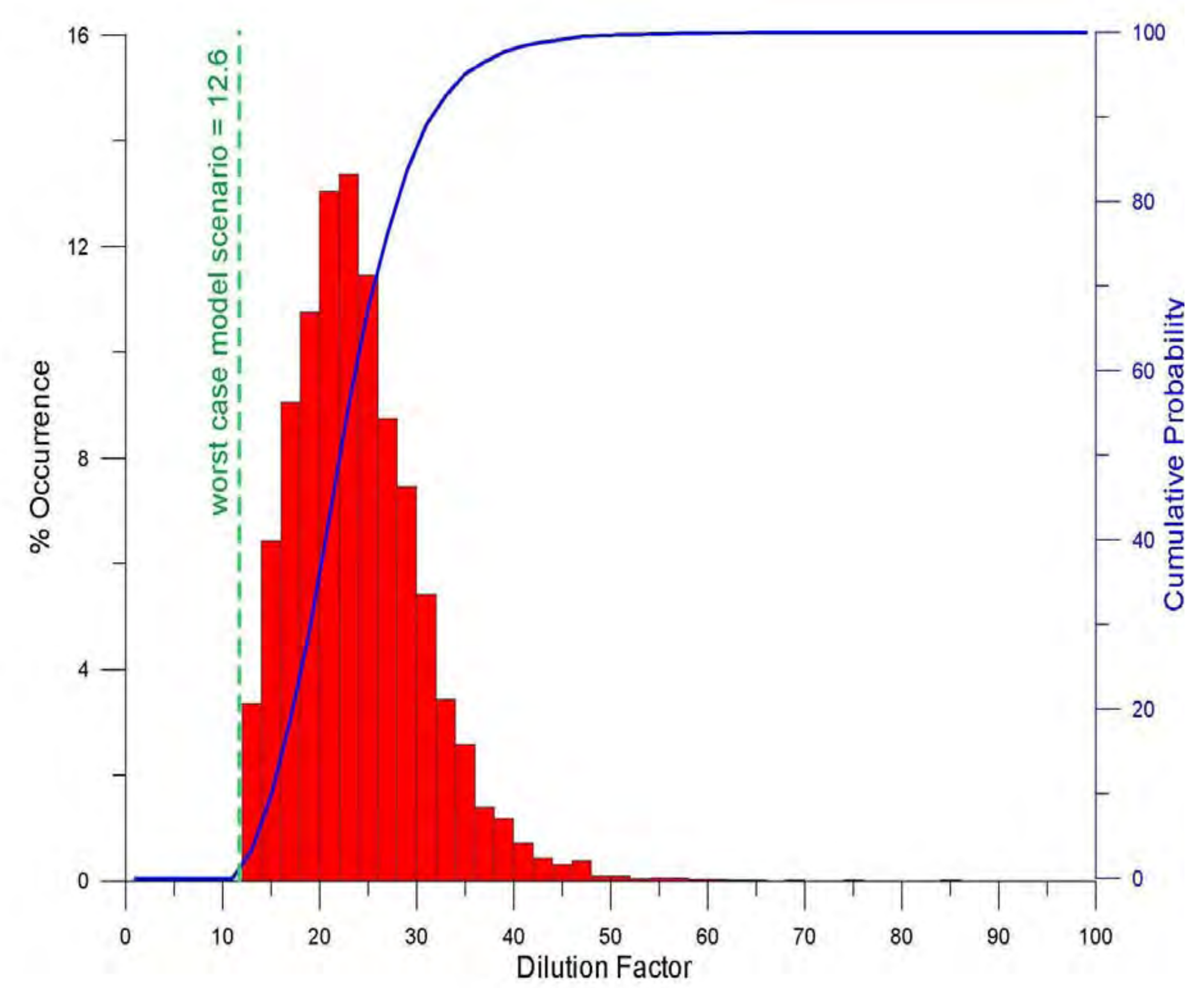


Figure 38. Probability density function (red) with corresponding cumulative probability (blue line) of the minimum surfzone dilution within the ZID around storm drain SDL-062. Derived from ocean boundary conditions and forcing functions, 1980-2012, 11,688 total numbers of realizations.

For comparison, probability statistics for minimum dilution are presented in Figure 39 at an offshore control point in ASBS 29 where local water depth is -10 m MSL (Figure-1). Here worst case minimum dilution of storm water is 13,200 to 1 while the median outcome from 11,688 model solutions is 52,000 to 1. The highest minimum dilutions of storm water at the offshore control point were found to be 4.2 x 10^5. Consequently, storm water dilution is sufficiently high in the offshore regions of ASBS 29 that concentrations of storm water runoff constituents should be well below quantifiable detection limits.

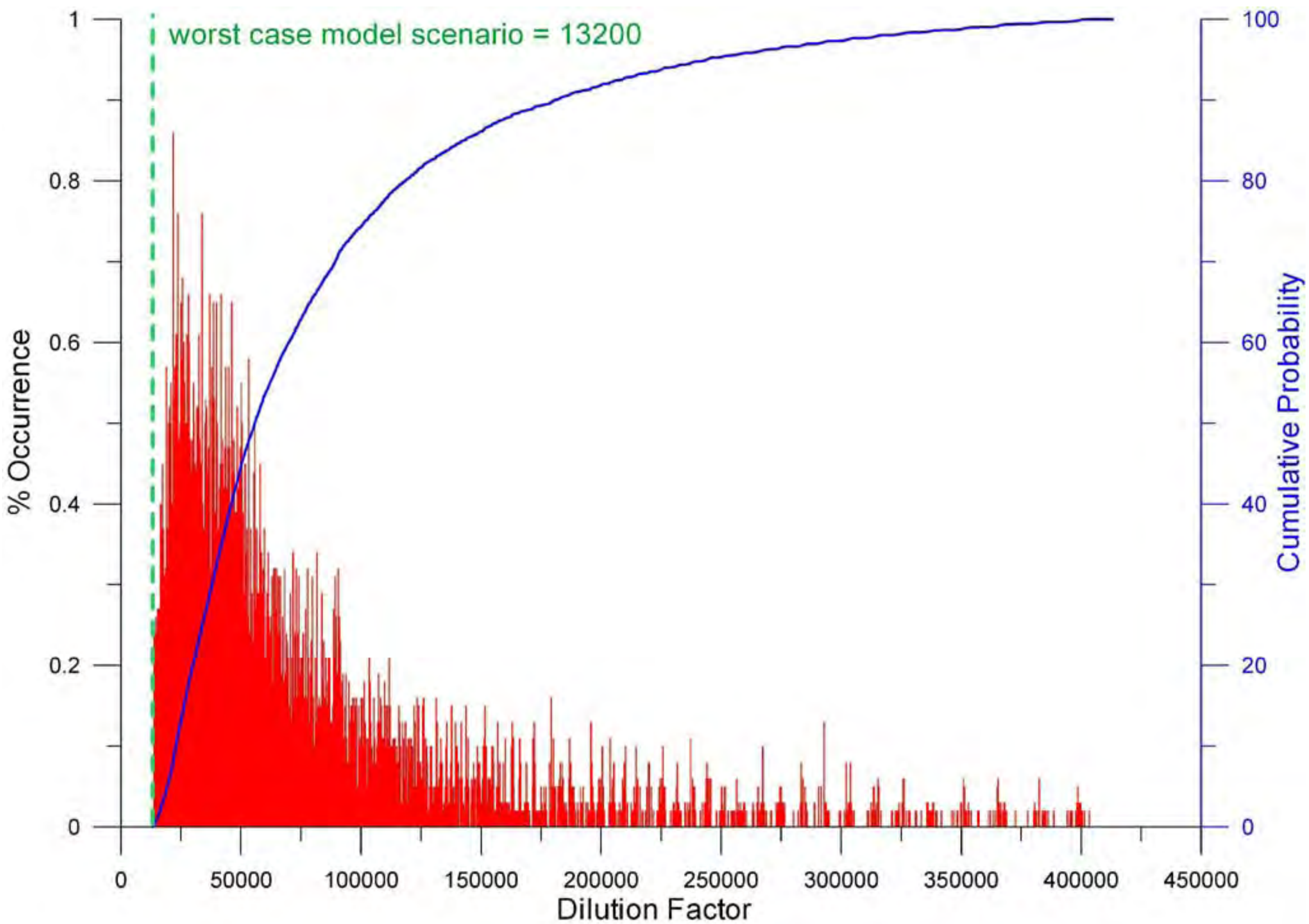


Figure 39. Probability density function (red) with corresponding cumulative probability (blue line) of the minimum dilution on the sea surface at offshore control point in ASBS 29. Derived from ocean boundary conditions and forcing functions, 1980-2012, 11,688 total numbers of realizations.

5.0 SUMMARY AND CONCLUSIONS:

During the 2011-2012 wet weather monitoring season (October 1, 2011 through April 30, 2012), storm water samples were collected from outfall discharge and receiving water mixing zone monitoring locations during three events (AMEC, 2012). These data were used to initialize the source loading inputs to a hydrodynamic model (***SEDXPORT***) used in the present receiving water analysis of storm water dilution and dispersion.

The ***SEDXPORT*** hydrodynamic modeling system was developed at Scripps Institution of Oceanography for the US Navy's *Coastal Water Clarity System* and *Littoral Remote Sensing Simulator*. This model has been peer reviewed multiple times and has been calibrated and validated in the Southern California Bight for six previous water quality and design projects.

Based on previous protocols established with the San Diego Regional Water Quality Control Board for storm drain dilution analysis in ASBS 31 (Jenkins and Wasyl, 2007), the numerical modeling study of the beach discharges is based on a wet weather worst-case scenario with a long-term probability assessment to bracket the envelope of the possible outcomes. Among the three wet-weather events monitored by AMEC (2012),Wet Weather Event-1 (20-21 November, 2011) comes closest to matching a computer search of marine environmental conditions for the wet-weather worst-case proxy. Analysis of the worst-case wet weather proxy was conducted over the entire coastal domain of La Jolla Bay and the Torrey Pines Littoral Sub-Cell.

The footprint of the dilution field for the wet weather worst case scenario, (involving the simultaneous discharge from storm drains SDL-157, SDL-062 and SDL-186), spreads about a kilometer seaward of the shoreline and about 2 kilometers along the shoreline, covering most of ASBS 29 and intruding into the southern portion of ASBS 31. The dilution factors of storm water in both ASBS 29 and ASBS 31 range between a minimum of 10^2 near shore to 10^7 along the seaward boundaries during the wet weather worst case. Dilution factors of 10^4 to 10^7 characterize the outer one-half of ASBS 29, while dilution factors 10^2 to 10^4 characterize the inner one-half. In the immediate neighborhood of SDL-157 and SDL-062, very near the shoreline of ASBS 29, dilution factors are as low as 40 to one.

The preponderance of the storm water discharge plume during the wet weather worst case scenario is from SDL-157 and SDL-062, while SDL-186 at Devil's Slide makes a very minor contribution. This is a fortuitous outcome in the sense that a significant amount of high-value, hard bottom marine habitat lives in the southern end of ASBS 29, and in the immediate neighborhood of SDL-186. Dilution factors of storm water discharged from SDL-186 are at most 10^3 near the shore in the nearfield of SDL-186, and more typically 10^4 to 10^5 elsewhere along the bluff-faced shoreline of the southern end of ASBS 29. The largest single contributing footprint to the combined storm water plume appears to be from SDL-062 at the end of Avendia De La Playa that produces a large patch with 10^2 minimum dilution very near the shoreline that spreads south along the shore to the La Jolla Beach and Tennis Club. The region influenced by this feature from SDL-062 is sandy, soft-bottom marine habitat. SDL-157 produces another patch with 10^2

minimum dilution, but this feature moves offshore and intrudes into the southern end of ASBS 31, where again the marine habitat is a sandy soft-bottom type.

Maximum concentrations of storm water born total suspended solids (TSS) in the receiving water due to SDL-186 storm water discharges are at most 0.001 mg/L near the shore in the nearfield of SDL-186, and more typically 0.0001 mg/L elsewhere along the bluff-faced shoreline of the southern end of ASBS 29, where significant high-value, hard bottom marine habitat resides. The highest TSS concentrations in the receiving water around SDL-062 (at the end of Avendia De La Playa), were found to be 3 mg/L in a nearshore patch that that spreads south along the shore to the La Jolla Beach and Tennis Club. Most of the shoreline impacted by SDL-157 experiences maximum TSS concentrations from storm water in the range of 0.3 mg/L to 0.03 mg/L, all of which is sandy soft-bottom marine habitat. At the northern end of ASBS 29, and extending into ASBS 31, maximum TSS concentrations in the receiving from SDL-157 are in the range of 4 mg/L to 0.4 mg/L, the highest anywhere in La Jolla Bay. These relatively higher TSS values are probably a consequence of the steep land forms that comprise the watershed of SDL-157, with portions of both developed and undeveloped coastal bluffs. Regardless, the marine habitat subjected to these TSS loadings from SDL-157 is a sandy, soft-bottom type.

Because of the disparity in length scales between dilution in the offshore region versus the surfzone, a separate analysis was performed on a nested fine scale grid covering the surf zone for a long shore reach that extended 1000 ft (305 meters) either side of the beach outfalls. The size of this grid was based on precedent already set for the definition of a zone of initial dilution (ZID) by the Regional Water Quality Control Board, San Diego. Thirty two years of receiving water variables (involving 11,688 distinct combinations) were input for daily simulations of dilution. The ZID was searched for the minimum dilution after averaging across the width of the surf zone. Source loading for these simulations were based on the discharges measured during Wet-Weather Event-1 from outfalls SDl-157, SDL-062 and SDL-186. The surfzone dilution results are then compared to minimum dilution on the sea surface at an offshore control point in ASBS 29 where local water depth is -10 m MSL.

This probability analysis procedure (based 11,688 historic combination of environmental variables) inevitably produces some combinations of wet weather discharges with dry weather ocean mixing conditions, and thereby results in certain worst case dilution scenarios that are more extreme, with even less dilution, than the wet weather worst case proxy derived from the monitoring program (Wet-Weather Event-1). For storm drain SDL-157, the median outcome for minimum dilution factor within the surfzone ZID is 32 to 1; however the potential range of minimum dilution goes as high as 88 to 1, and as low as 15 to 1. Low energy, dry weather ocean mixing produced the 15 to 1 minimum surfzone dilution outcome, which had a probability of occurrence of 0.13%. Altogether, 90% of the potential outcomes produce minimum dilutions in the surfzone ZID greater than 20 to 1.

Minimum surfzone dilutions in the ZID off SDL-062 are slightly less. This is due to the wave shadow found chronically in the refraction pattern of La Jolla Shores in the neighborhood of Avendia De La Playa. The minimum surfzone dilution of discharges from SDL-062 is found to be 12.6 to 1, again a low energy, dry weather ocean mixing result. The median outcome of minimum surfzone dilution for SDL-062 is 22 to 1; while the potential range of minimum dilution goes as high as 64 to 1, based on the 32-year historical sequence of receiving water variables. Ninety percent of the potential outcomes for SDL-062 produce minimum surfzone dilutions greater than 15 to 1.

For comparison, probability statistics for minimum dilution at an offshore control point in ASBS 29 (where local water depth is -10 m MSL) discovered a worst case minimum dilution of storm water of 13,200 to 1; while the median outcome is 52,000 to 1. The highest minimum dilutions of storm water at the offshore control point were found to be 4.2×10^5. Consequently, storm water dilution is sufficiently high in the offshore regions of ASBS 29 that concentrations of storm water runoff constituents should be well below quantifiable detection limits.

6.0 REFERENCES AND BIBLIOGRAPHY

Armi, L. A. (1979). *Effects of variations in eddy diffusivity on property distributions in the oceans*, Jour. of Mar. Res., v. 37, n. 3, p. 515-530.

Berkoff, J. C. W. (1972). *Computation of combined refraction-diffraction*, Proc. 13th Coastal Eng. Conf., p. 471-490.

Boas, M. L. (1966). *Mathematical Methods in the Physical Sciences*, John Wiley & Sons, Inc., New York, 778 pp.

Coastal data information program (CDIP) (2001). University of California, San Diego (UCSD) Scripps Institution of Oceanography (SIO) Reference Series, 01-20 and http://cdip.ucsd.edu.

Connor, J. J. and Wang, J.D. (1973). *Finite element modeling of two-dimensional hydrodynamic circulation*, MIT Tech Rpt., #MITSG 74-4, p. 1-57.

Conte, S. D. and DeBoor, C. (1972). *Elementary Numerical Analysis*, Second Edition, McGraw-Hill Book Co., New York.

Cox, R. A., and N. D. Smith (1959). *The specific heat of sea water*, Proc. Roy. Soc., A, 252, 51-62.

Dalrymple, R. A., Kirby, J.T. and Hwang, P.A. (1984). *Wave diffraction due to areas of energy dissipation*, Jour. Waterway Port, Coast, and Ocean Engineering, v. 110, p. 67-79.

DeGroot, S. R. and Mazur, P. (1984). *Non-Equilibrium Thermodynamics*, Dover Pub., Inc., New York, 510 pp.

Durst, C. S. (1924). *The relationship between current and wind*, Quart. J. R. Met. Soc., v. 50, p. 113 (London).

Finlayson, B. A. (1972). *The Method of Weighted Residuals and Variational Principles*, Academic Press.

Flick, R.E. and Sterrett, E.H. (1994). *The San Diego Shoreline, Shoreline Erosion Assessment and Atlas of the San Diego Region*, Vol. 1, California Department of Boating and Waterways, 135 pp.

Fujita, H. (1962). *Mathematical Theory of Sedimentation Analysis*. York: Academic, 315pp.

Gallagher, R. H. (1981). *Finite Elements in Fluids*, John Wiley & Sons, New York, 290 pp.

Goddard, L. and Graham, N.E. (1997). *El Niño in the 1990's*, Jour. Geophysical Res., v. 102, n. C5, p. 10,423-36.

Inman, D. L., Nordstorm, C. E. and Flick, R. E. (1976). *Currents in submarine canyons: an air-sea-land interaction*, p. 275-310 in M. Van Dyke, et al (eds.), Annual Review of Fluid Mechanics, v. 8, 418 pp.

Inman, D. L., Nordstorm, C. E. and Tait, R. J. (1971). *Mixing in the surf zone* Jour. Geophys. Res., v.76, no. 15, pp 3493-3514.

Inman, D. L., Flick, R. E. and Guza, R. T. (1981). *Elevation and velocity measurements of laboratory shoaling waves*, J. Geophys. Res., 86(C5), 4149-4160.

Inman, D. L. and Masters, P. M. (1991a). *Coastal sediment transport concepts and mechanisms*, Chapter 5 (43 pp.) in *State of the Coast Report, San Diego Region, Coast of California Storm and Tidal waves Study*, U. S. Army Corps of Engineers, Los Angeles District Chapters 1-10, Appen. A-I, 2 v.

Inman, D. L. and Masters, P. M., (1991b). *Budget of sediment and prediction of the future state of the coast*, Chapter 9 (105 pp.) in *State of the Coast Report, San Diego Region, Coast of California Storm and Tidal Waves Study*, U. S. Army Corps of Engineers, Los Angeles District, Chapters 1-10, Appen. A-I; 2 v.

Inman D. L. and Jenkins, S. A. (2006). *Thermodynamic solutions for equilibrium beach profiles*, Jour. Geophys. Res., v.3, C02003, doi:10.1029, 21pp.

Inman, D. L. and Jenkins, S. A. (1999). *Climate change and the episodicity of sediment flux of small California rivers*, Jour. Geology, v. 107, p. 251–270. http://repositories.cdlib .org/sio/cmg/2/

Inman, D. L. and Jenkins, S. A. (1997). *Changing wave climate and littoral drift along the California coast*, p. 538-49 in O. T. Magoon et al., eds., California and the World Ocean '97, American Society of Civil Engineers (ASCE), Reston, VA, 1756 pp.

Inman, D. L. and Jenkins, S. A. (1996). *A chronology of ground mine studies and scour modeling in the vicinity of La Jolla*, UCSD SIO Reference Series 96-13, 26 pp.

Inman, D. L. and Jenkins, S. A. (1985). *Erosion and accretion waves from Oceanside Harbor*, p. 591-3 in Oceans 85: Ocean Engineering and the Environment, Marine Technological Society & IEEE, v. 1, 674 pp.

Inman D. L. and Jenkins, S. A. (1985). *On a submerged sphere in a viscous fluid excited by small amplitude periodic motion*, Jour. Fluid Mech., v. 157, p. 199-124.

Inman, D. L., Jenkins, S. A. and Elwany, M. H. S. (1996). *Wave climate cycles and coastal engineering practice*, Coastal Engineering, 1996, Proc. 25th Int. Conf., (Orlando), ASCE, New York, v. 1, Ch 25, p. 314-27.

Inman, D. L., Jenkins, S. A. and Elwany, M. H. S. (1993). *Shorerise and bar-berm profiles on ocean beaches*, Jour. Geophys. Res., 98(C10), p. 18, 181-199.

Inman, D. L., Jenkins, S. A. and Van Dorn, W. G. (1981). *Evaluation of sediment management procedures*, UCSD SIO Reference Series No. 81-22, 212 pp.

Inman D. L., Jenkins, S. A., Wasyl, J., Richardson, M. D., Wever, T.F. (2007). *Scour and burial mechanics of objects in the nearshore*, IEEE Jour.Oc.Eng, vol. 32, no. 1, pp 78-90.

Inman D. L. and Komar, P. D. (1970). *Longshore sand transport of beaches*, Jour. Geophys. Res., v. 75, n. 30, p. 5914-5927.

Inman, D. L., Pawka, S. S., Lowe, R. L., and Holmes, L. (1976). *Wave climate at Torrey Pines Beach, California*, U. S. Army Corps of Engineers, Coastal Engineering Research Center, Tech Paper 76-5, 372 pp.

Jenkins, S. A., Paduan, J., Roberts, P., Schlenk, D., and Weis, J. (2012). *Management of Brine Discharges to Coastal Waters; Recommendations of a Science Advisory Panel*, submitted at the request of the California Water Resources Control Board, 56 pp. + App.

Jenkins, S. A. and Skelly, D. W. (1989). *An evaluation of the coastal data base pertaining to seawater diversions at Encina Power Plant*, UCSD SIO Reference Series, #89-4, 52 pp.

Jenkins, S. A. and Wasyl, J. (2007). *Hydrodynamic Modeling of Shoreline Discharges of Laboratory Seawater and Storm Water at Scripps Beach, CA*, submitted to UCSD Facilities Design and Construction, 266 pp.

Jenkins, S. A. and Wasyl, J. (2005). *Coastal evolution model*, UCSD SIO Tech. Rpt. No. 58, 179 pp + appendices. http://repositories.cdlib.org/sio/techreport/58/

Jenkins, S. A. and Wasyl, J. (1990). *Resuspension of estuarial sediments by tethered wings*, Jour. of Coastal Res., v. 6, n. 4, p. 961-980.

Jenkins, S. A., Wasyl, J. and Aijaz, S. (1992). *Transport of fine sediment by hydrostatic jets*, Coastal and Estuarine Studies, American Geophysical Union, v. 42, p. 331-47.

Jenkins, S. A., Wasyl, J. and Jenkins, S. A. (1989). *Dispersion and momentum flux study of the cooling water outfall at Agua Hedionda*, UCSD SIO Reference Series, #89-17, 36 pp.

Jenkins, S. A., Wasyl, J., Cleveland, J. S., Talcott, J.C., Health, A.L., Goosby, S.G., Schmitt, K.F., and Leven, L.A. (1995). *Coastal water clarity modeling*, Science Applications International Corporation (SAIC), Tech. Rpt. 01-1349-03-4841-000, 491 pp.

Jerlov, N.G. (1976). *Marine Optics*, Elsevier, Amsterdam, 231 pp.

Kirby, J. T. (1986a). *Higher-order approximations in the parabolic equation method for water waves*, Jour. Geophys. Res., v. 91, C1, p. 933-952.

Kirby, J. T. (1986b). *Rational approximations in the parabolic equation method for water waves*, Coastal Engineering, 10, p. 355-378.

Kirby, J. T. (1986c). *Open boundary condition in the parabolic equation method*, Jour. Waterway, Port, Coastal, and Ocean Eng., 112(3), p. 460-465.

Krone, R.B. (1962). *Flume Studies of the Transport of Sediment in Estuarial Shoaling Processes*, University of California, Berkeley Press, 69 pp.

Lazara, B. J. and Lasheras, J. C. (1992a), *Particle dispersion in a developing free shear layer, Part 1, Unforced flow*, Jour. Fluid Mech. 235, p. 143-178.

Lazara, B. J. and Lasheras, J. C. (1992b). *Particle dispersion in a developing free shear layer, Part 2, Forced Flow*, Jour. Fluid Mech., 235, p. 179-221.

LePage, S. and Ware, R. (2001). *Assessment of biological impacts of the Agua Hedionda jetty restoration project*, Tech. Rpt. Submitted to Cabrillo Power 1LLC.

List, E. J., Gartrell, G. and Winant, C. D. (1990). *Diffusion and dispersion in coastal waters*, Jour. Hydraulic Eng., v. 116, n. 10, p. 1158-79.

Longuet-Higgins, M. S. (1970). *Longshore currents generated by obliquely incident waves*, Jour. Geophys. Res., v. 75, n. 33, p. 6778-6789.

MACTEC. (2011a). *La Jolla Shores Area of Special Biological Significance (ASBS) Compliance Monitoring Work Plan*. October 2011. Prepared for the City of San Diego

MACTEC. (2011b). *La Jolla Shores Area of Special Biological Significance (ASBS) Compliance Monitoring Work Plan Addendum*. November 2011. Prepared for the City of San Diego

Martin, J. E. and Meiberg, E. (1994). *The accumulation and dispersion of heavy particles in forced two-dimensional mixing layers, 1: The fundamental and subharmonic cases*, Phys. Fluids, A-6, p. 1116-1132.

National Oceanic and Atmospheric Administration (NOAA) (2000). *Verified/Historical Water Level Data*, http://www.opsd.nos.noaa.gov/data_res.html

Neumann, G. (1952). *Ober die komplexe Natur des Seeganges, Teil 1 and 2*, Deut. Hydrogr. Zeit., v. 5, n. 2/3, p. 95-110, n. 5/6, p. 252-277.

Neumann, G., Pierson, Jr., W. J. (1966). *Principles of Physical Oceanography*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 545 pp.

Nielsen, P. (1979). *Some basic concepts of wave sediment transport*, Series Paper No. 20, Institute of Hydrodyn. and Hydro. Eng., Tech. Univ. of Denmark.

Oden, J. T. and Oliveira, E. R. A. (1973). *Lectures on Finite Element Methods in Continuum Mechanics*, The University of Alabama Press.

Pawka, S. S. (1982). *Wave directional characteristics on a partially sheltered coast*, PhD dissertation, UCSD, 246 pp.

PBS&J (2005). UCSD/SIO seawater separation and discharge" Appendix C of Jacobs and Associates design submission to UCSD Facilities, Design and Construction May 20, 2005, 10 pp + figs.

Radder, A. C. (1979). *On the parabolic equation method for water-wave propagation*, J. Fluid Mech., 95, part 1, p. 159-176.

Schmidt, W. (1917). *Wirkungen der ungeordneten Bewegungen im Wasser der Meere und Seen*, Ann. D. Hydr. u. Marit. Meteorol., vol. 45, p. 367-381.

Schoonmaker, J. S., Hammond, R. R., Heath, A. L., and Cleveland, J. S. (1994). *A numerical model for prediction of sub-littoral optical visibility*, SPIE Ocean Optics XII, 18 pp.

Scripps Institution of Oceanography (SIO) (2001). *SIO shore station, Scripps Pier*, http://www-mlrg.ucsd.edu/shoresta/mnSIOMain/siomain.htm

Stommel, H. (1949). *Horizontal diffusion due to oceanic turbulence*, Journal of Marine Research, v. VIII, n. 3, p. 199-225.

Thorade, H. (1914). *Die Geschwindigkeit von Triftstormungen und die Ekman'sche Theorie*, Ann. D Hydr. u. Marit. Meteorol., v. 42, p. 379.

University of California, San Diego (UCSD) (2005a). *Quarterly (January – March, 2005) Monitoring Report - Attachment C*, Discharge Order No. R9-2005-0008, NPDES Permit No. CA0107239. UCSD SIO. June 1, 2005.

UCSD (2005b). *Quarterly (April – June, 2005) Monitoring Report -. Attachment C*, Discharge Order No. R9-2005-0008, NPDES Permit No. CA0107239. UCSD SIO. Volume I. September 1, 2005.

UCSD, (2005c). *Quarterly (January – March, 2005) Monitoring Report - Attachment B*, Discharge Order No. R9-2005-0008, NPDES Permit No. CA0107239. UCSD SIO. June 1, 2005.

UCSD, (2005d). *Quarterly (April – June, 2005) Monitoring Report - Attachment B*, Discharge Order No. R9-2005-0008, NPDES Permit No. CA0107239. UCSD SIO. Volume I. September 1, 2005.

UCSD, (2005e). *Semi-Annual (January – June, 2005) Monitoring Report*, Discharge Order No. R9-2005-0008, NPDES Permit No. CA0107239. UCSD SIO. Volume II. September 1, 2005.

Weston Solutions, Inc. (2011). *La Jolla Shores Area of Special Biological Significance Regional Compliance Monitoring, 2010-2011, Final Report*. June, 2011. Prepared for the City of San Diego.

Weston Solutions, Inc. (2010). *La Jolla Shores Area of Special Biological Significance Regional Compliance Monitoring, 2009-2010, Final Report*. June, 2010. Prepared for the City of San Diego.

Weston Solutions, Inc. (2009). *La Jolla Shores ASBS Regional Compliance Monitoring, 2008-2009, Final Report*. June, 2009. Prepared for the City of San Diego.

Weston Solutions, Inc. (2007). *La Jolla Shores Watershed Urban Runoff Characterization and Watershed Characterization Study, Final Report*. July, 2007. Prepared for the City of San Diego.

Wang, H. P. (1975). *Modeling an ocean pond: a two-dimensional, finite element hydrodynamic model of Ninigret Pond, Charleston, Rhode Island*, Univ. of Rhode Island, Marine Tech. Rpt., #40, p. 1-58.

Weiyan, T. (1992). *Shallow Water Hydrodynamics*, Water & Power Press, Hong Kong, 434 pp.

White, W. B. and Cayan, D.R. (1998). *Quasi-periodicity and global symmetries in interdecadal upper ocean temperature variability*, Jour. Geophysical Res., v. 103, n. C10, p. 21,335-54.

Winant, C. D. (1974). *Internal surges in coastal waters*, Jour. Geophys. Res., v. 79, n. 30, p. 4523-26.

APPENDIX A

GOVERNING EQUATIONS AND CODE FOR THE TIDE_FEM CURRENT MODEL

Appendix A:
Governing Equations and Code for the TIDE_FEM Current Model

A finite element approach was adapted in preference to more common finite difference shallow water tidal models, e.g., Leendertse (1970), Abbott et al (1973), etc. Finite difference models employ rectangular grids which would be difficult to adapt to the complex geometry of the La Jolla Bay. It is believed that large errors would accumulate from attempting to approximate the irregular boundaries of the La Jolla Bay system with orthogonal segments. On the other hand, finite element methods allow the computational problem to be contained within a domain bounded by a continuous contour surface, such as the S_f contours stored within the *bathy* bathymetry file.

A finite element tidal hydraulics model, **TIDE_FEM**, [Inman and Jenkins, 1996] was employed to evaluate the tidal hydraulics of the La Jolla Bay (containing ASBS 31). **TIDE_FEM** was built from some well-studied and proven computational methods and numerical architecture that have done well in predicting shallow water tidal propagation in Massachusetts Bay [Connor and Wang, 1974] and estuaries in Rhode Island, [Wang, 1975], and have been reviewed in basic text books [Weiyan, 1992] and symposia on the subject, e.g., Gallagher (1981).

TIDE_FEM employs a variant of the vertically integrated equations for shallow water tidal propagation after Connor and Wang (1975). These are based upon the Boussinesq approximations with Chezy friction and Manning's roughness. The finite element discretization is based upon the commonly used ***Galerkin weighted residual method*** to specify integral functionals that are minimized in each finite element domain using a variational scheme, see Gallagher (1981). Time integration is based upon the simple ***trapezoidal rule*** [Gallagher, 1981]. The computational architecture of **TIDE_FEM** is adapted from Wang (1975), whereby a transformation from a **global** coordinate system to a **natural** coordinate system based on the unit triangle (see Figure A-1) is used to reduce the weighted residuals to a set of order-one ordinary differential equations with constant coefficients. These coefficients (***influence coefficients***) are posed in terms of a ***shape function*** derived from the natural coordinates of each nodal point. The resulting systems of equations are assembled and coded as banded matrices and subsequently solved by ***Cholesky's method***, see Oden and Oliveira (1973 and Boas (1966).

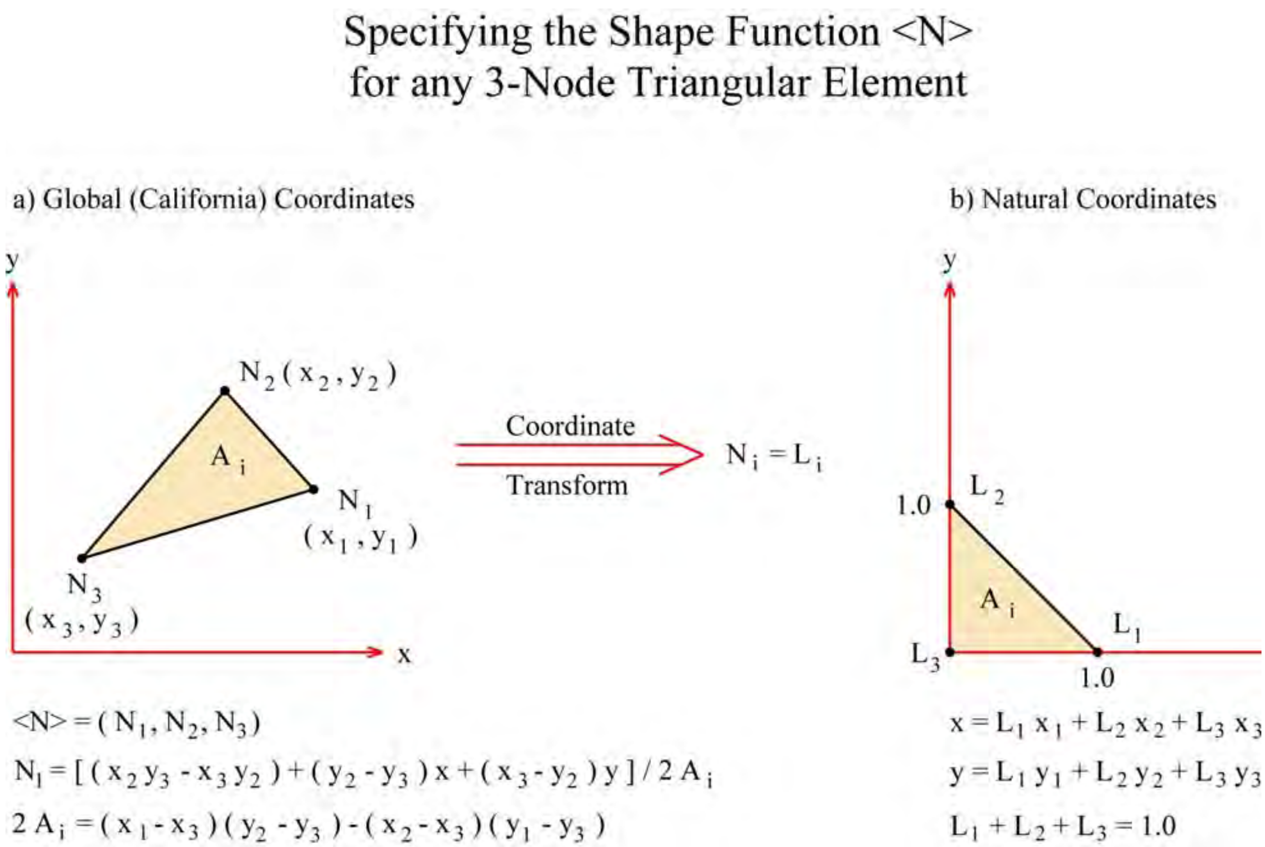


Figure A-1. Shape function polynomial and transform to natural coordinates for a generalized 3-node triangular element in the cross-shore plane

We adapt the California coordinates as our **global** coordinate system (x, y) to which the nodes in the computational mesh are referenced, with **x** (easting) and y (northing). The vertical coordinate z is fixed at 0.0 ft NGVD and is positive upward. The local depth relative to 0.0 ft NGVD is **h** and the mean surface elevation about 0.0 ft NGVD is **η**. The total depth of water at any node is **H = h + η**. The vertically averaged xy-components of velocity are $(\bar{u}, \bar{v})$.The continuity and momentum equations may be written from Connor and Wang, (1974), as:

$$\frac{\partial}{\partial t}\rho H + \frac{\partial}{\partial x}q_x + \frac{\partial}{\partial y}q_y = 0$$

$$\frac{\partial}{\partial t}q_x + \frac{\partial}{\partial x}\bar{u}q_x + \frac{\partial}{\partial y}\bar{u}q_y = B_x + \frac{\partial}{\partial x}\left(F_{xx} - F_p\right) + \frac{\partial}{\partial x}F_{yx}$$

$$\frac{\partial}{\partial t}q_y + \frac{\partial}{\partial x}\bar{v}q_x + \frac{\partial}{\partial y}\bar{v}q_y = B_y + \frac{\partial}{\partial y}\left(F_{yy} - F_p\right) + \frac{\partial}{\partial x}F_{xy} \qquad \text{(A2)}$$

Here $\mathbf{q_x}$, $\mathbf{q_y}$ are mass flux components

$$q_x = \rho \int_{-h}^{\eta} \bar{u} dz \quad \text{(A3)}$$

$$q_y = \rho \int_{-h}^{\eta} \bar{v} dz \quad \text{(A4)}$$

and $\mathbf{q_I}$ is the mass flux through the ocean inlet due to water surface elevation changes in the estuary:

$$q_I = \rho \frac{\partial}{\partial t}\left(\frac{\partial s}{\partial \eta}\right) \quad \text{(A5)}$$

$\mathbf{F_p}$ is the pressure force resultant and $\mathbf{F_{xx}}$, $\mathbf{F_{xy}}$, $\mathbf{F_{yy}}$ are "equivalent" internal stress resultants due to turbulent and dispersive momentum fluxes

$$\begin{aligned} F_p &= \int_{-h}^{\eta} p dz = \frac{\rho g H^2}{2} \\ F_{xx} &= 2\varepsilon \frac{\partial}{\partial x} q_x \\ F_{yy} &= 2\varepsilon \frac{\partial}{\partial y} q_y \\ F_{yx} = F_{xy} &= \varepsilon\left(\frac{\partial}{\partial y} q_y + \frac{\partial}{\partial x} q_x\right) \end{aligned} \quad \text{(A6)}$$

and ε is the eddy viscosity. $\mathbf{B_x}$ and $\mathbf{B_y}$ are the bottom stress components

$$\begin{aligned} B_x &= \tau_x + \rho g H \frac{\partial h}{\partial x} \\ B_y &= \tau_y + \rho g H \frac{\partial h}{\partial y} \end{aligned} \quad \text{(A7)}$$

In Equation (A7), $\boldsymbol{\tau_x}$ and $\boldsymbol{\tau_y}$ are the bottom shear stress components that are quasi-linearized by Chezy-based friction using Manning's roughness factor, $\mathbf{n_o}$:

$$\tau_x = -\frac{g}{\rho H^2 C_z^{\ 2}} q_x \left(q_x^{\ 2} + q_y^{\ 2}\right)^{1/2}$$
$$\tau_y = -\frac{g}{\rho H^2 C_z^{\ 2}} q_y \left(q_x^{\ 2} + q_y^{\ 2}\right)^{1/2} \qquad \text{(A8)}$$

where $\mathbf{C_z}$ is the Chezy coefficient calculated as:

$$C_z = \frac{1.49}{n_0} H^{1/6} \qquad \text{(A9)}$$

Boundary conditions are imposed at the locus of possible land/water boundaries, $\mathbf{S_f}$ in the ***bathym*** file and at the shoreline, $\mathbf{S_o}$. Flux quantities normal to these contours are denoted with "n" subscripts and tangential fluxes are given "s" subscripts. At any point along a boundary contour, the normal and tangential mass fluxes are:

$$q_n = \int_{-h}^{\eta} \rho u_n dz = \alpha_{nx} q_x + \alpha_{ny} q_y$$
$$q_s = \int_{-h}^{\eta} \rho u_s dz = -\alpha_{nx} q_x + \alpha_{ny} q_y$$
$$\alpha_{nx} = \cos(n, x)$$
$$\alpha_{ny} = \cos(n, y) \qquad \text{(A10)}$$

Components of momentum fluxes across a boundary are equivalent to internal force resultants according to:

$$F_{nx} = \alpha_{nx}(F_{xx} - F_p) + \alpha_{ny} F_{yx}$$
$$F_{ny} = \alpha_{ny}(F_{yy} - F_p) + \alpha_{nx} F_{xy} \qquad \text{(A11)}$$

On land boundary contours, the flux components are prescribed

$$q_n = q_s = 0 \qquad \text{on land} \qquad \text{(A12)}$$

On the deep water ocean boundary, the normal boundary forces (due to sea surface elevation) are continuous with ocean values, and the mass exchange is limited by the storage capacity of the estuary. Hence

$$F_{nm} = \overline{F}_{nm} \quad \text{and} \quad q_{nm} = q_I \quad \text{ocean boundary} \qquad \text{(A13)}$$

In the problem at hand $\overline{F}_{nn}$ is prescribed on the shoreline by the ocean tidal elevation, η_0, and the local depth, $\mathbf{h_o}$ according to

$$\overline{F}_{nm} = \frac{\rho g}{2}\left(\eta_0 + h_0\right)^2 \quad \text{on S0} \qquad \text{(A14)}$$

Ocean ***tidal forcing functions*** η_0 were developed in Section 3.1. The ocean boundary condition as specified by Equation (A13) places a dynamic boundary condition on the momentum equations and a kinematic boundary condition on the continuity equation that is constrained by the storage rating curve. Solutions are possible by specifying only the dynamic boundary condition, but then mass exchanges are controlled by the wetting and drying of individual grid cells with associated discretization and interpolation errors which threaten mass conservation. The technique of over specifying the ocean boundary condition with both a dynamic and kinematic condition is discussed in the book by Weiyan (1992).

The governing equations (A2-A9) and the boundary conditions (A10-A13) are cast as a set of integral functionals in a variational scheme, [Boas, 1966]. Within the domain of each element of the mesh, $\mathbf{A_i}$ the unknown solution to the governing equations is simulated by a set of ***trial functions*** $(\hat{H}, \hat{q})$ having adjustable coefficients. The trial functions are substituted into the governing equations to form ***residuals***, ($\mathbf{R_H}$, $\mathbf{R_q}$). The residuals are modified by ***weighting functions***, (**ΔH, Δq**). The coefficients of the trial functions are adjusted until the weighted residuals vanish. The solution condition on the weighted residuals then becomes:

$$\iint_{A_i} R_H \Delta H dA = 0$$

$$\iint_{A_i} R_q \Delta q dA = 0$$

By the Galerkin method of weighted residuals, [Finlaysen, 1972], the weighting functions are set equal to nodal ***shape functions***, <**N**>, or:

$$\Delta H \sim N_i$$
$$\Delta q \sim N_j$$

The shape function, <N>, is a polynomial of degree which must be at least equivalent to the order of the highest derivative in the governing equations. The shape function also provides the mechanism to discretize the governing equations. Figure A-1 gives the shape function polynomial in terms of ***global (California)*** coordinates for the first nodal point, $\mathbf{N_1}$ of a generalized 3-node triangular element of area $\mathbf{A_i}$. Wang (1975) obtained significant numerical efficiency in computing the weighted residuals when the shape functions of each nodal point, $\mathbf{N_i}$, are transformed to a system of ***natural*** coordinates based upon the unit triangle, giving $\mathbf{N_i} \rightarrow \mathbf{L_i}$ as detailed in Figure A-1. The shape functions also permit semi-discretization of the governing equations when the trial functions are posed in the form:

$$\hat{H}(x,y,t) = \sum_i H_i(t) N_i(x,y)$$
$$\hat{q}(x,y,t) = \sum_j q_j(t) N_j(x,y) \qquad \text{(A15)}$$

Discretization using the weighting and trial functions expressed in terms of the nodal shape functions allows the ***distribution*** of dependent variables over each element to be obtained from the values of the independent variables at discrete nodal points. However, the shape function at any given nodal point, say $\mathbf{N_1}$, is a function of the independent variables of the two other nodal points which make up that particular 3-node triangular element, see Figure A-1. Consequently, the computations of the weighted residuals lead to a series of influence coefficient matrices defined by

$$a_{ij} = \frac{1}{A_i}\iint N_i N_j \, dA$$
$$s_{ij} = \frac{1}{A_i}\iint N_i \frac{\partial N_j}{\partial x} dA$$
$$t_{ij} = \frac{1}{A_i}\iint N_i \frac{\partial N_j}{\partial y} dA$$
$$g_{ijk} = \frac{1}{A_i}\iint N_i N_j \frac{\partial N_k}{\partial x} dA$$
$$h_{ijk} = \frac{1}{A_i}\iint N_i N_j \frac{\partial N_k}{\partial y} dA \qquad \text{(A16)}$$

The influence coefficient matrices given by equation (A16) are evaluated in both global and natural coordinates. Once the influence coefficients have been calculated for each 3-node element, the weighted residuals reduce to a set of order-one ordinary differential with constant coefficients. The continuity equation becomes:

$$\sum \left(a_{ij} \frac{dH_i}{dt} \right) = -\sum_i \sum_k \left[g_{ijk} \left(H_i q_{xk} + H_k q_{xi} \right) + h_{ijk} \left(H_i q_{yk} + H_k q_{yi} \right) \right]$$
$$\sum \left(a_{ij} \frac{dq_{xj}}{dt} \right) = -\sum_j \sum_k \left[g_{ijk} \left(q_{xk} q_{xj} \right) + h_{ijk} \left(q_{yj} q_{xk} \right) \right] + N_i \sum_j N_j S_{fj} + g \sum_i s_{ij} H_i$$
$$\sum \left(a_{ij} \frac{dq_{yj}}{dt} \right) = -\sum_j \sum_k \left[g_{ijk} \left(q_{xj} q_{yk} \right) + h_{ijk} \left(q_{yj} q_{xk} \right) \right] + N_i \sum_j N_j S_{fj} + g \sum_i t_{ij} H_i \qquad \text{(A17)}$$

Equations (A17) are essentially simple oscillator equations forced by the collection of algebraic terms appearing on the right hand side; and are therefore easily integrated over time. The time integration scheme used over each time step of the tidal forcing function is based upon the ***trapezoidal rule***, see Gallagher 1981) or Conte and deBoor (1972). This scheme was chosen because it is known to be unconditionally stable, and in tidal propagation problems has not been known to introduce spurious phase differences or damping. It replaces time derivatives between two successive times, $\Delta t = t_{n+1} - t_n$, with a truncated Taylor series. For the water depth it would take on the form:

$$\frac{dH}{dt} = \eta(t)$$
$$H_{n+1} - H_n = \frac{\Delta t}{2}\left(\eta_{n+1} + \eta_n\right) + E\Delta t$$
$$E = \frac{1}{12}\left(\Delta t\right)^2 \left|\frac{d^2\eta}{dt^2}\right| \qquad \text{(A18)}$$

To solve equation (A18), iteration is required involving successive forward and backward substitutions.

Of particular interest are the ***hydraulic friction slope coefficients, S_{fj},*** appearing as damping terms on the right hand side of equation (A17). Two separate formulations are used. One is given for the 3-node triangular elements situated in the interior of La Jolla Bay which do not experience successive wetting and drying during each tide cycle. The other formulation is used for the hydraulic friction slope coefficients for the elements situated along the wet and dry boundaries of the shoreline. These have been formulated as 3-node triangular elements with one curved side based upon the cubic-spline matrices developed by Weiyan (1992). The wet-dry boundary coordinates of the curved side, (x=, y=), are linearly interpolated for any given water elevation from the contours stored in the ***bathym*** file.

The influence and friction slope coefficient matrices together with the trapezoidal rule reduce equations (A16) and (A17) to a system of algebraic equations [Grotkop, 1973] which are solved by Cholesky's method per a numerical coding scheme by Wang (1975). For more details, refer to the TIDE_FEM code below, and Gallagher (1981) or Oden and Oliveira (1973).

Bathymetry errors are the most common cause of modeling errors. Other sources of errors include:

ELEMENT INTERPOLATION ERROR**:** Due to the degree of the polynomial used to specify shape function, N_i.

DISCRETIZATION ERRORS**:** Due to mesh coarseness and approximating the curved wet/dry boundary side of an element with a quadratic spline.

***QUADRATURE ERRORS*:** Due to reducing the weighted residual integral with the influence coefficient matrices.

***ITERATION ERRORS*:** Due to solving the system of algebraic equations reduced from the Galerkin Equations.

ROUNDOFF ERRORS: Due to time integration by the trapezoidal rule.

***SEA LEVEL ANOMALIES*:** Due to discrepancies between the astronomic tides and the actual observed water levels in the ocean.

***INSUFFICIENT CALIBRATION DATA*:** Due to limitations in the period of record

The following is a listing of the TIDE_FEM code:

```
C..tide_fem.f
c..Finite element tidal hydraulics model using the Galerkin weighted
c..residual process to implement the finite element scheme
c..with Chezy based friction terms.
c..Adapted from Wang (1975) by Scott Jenkins and Joseph Wasyl.
c..7/16/04
c
      dimension X(500), Y(500), H(500),B1(1000,100),B2(500,50),
     & U(500), V(500), ETA(500), UVP(500), EP(500),MEXT(500),
     & MINT(500), R1(1000),R2(500),A(20,1000),Q(1000),MB(500),
     & CMAN(1000),CHEZ(1000),AREA(1000),N1(1000),N2(1000),N3(1000),
     & MH(10),MLE(200),ANG(200),tide(500),time(500)
      character*12 tidein, bathym, nodes, datout, temp
c
      open(20,file='tide_fem.inp',status='old')
      read(20,'(a12)')tidein
      read(20,'(a12)')bathym
      read(20,'(a12)')nodes
      read(20,'(a12)')temp
      read(20,'(a12)')datout
c
C..READ COMPUTATIONAL PARAMETERS
      READ(20,280) NN,NM,NW
280   FORMAT(3I10)
      READ(20,281) T,TLIM,DT,WW
281   FORMAT(4F10.0)
      READ(20,282) ATTD,HA,WX,WY
282   FORMAT(4F10.1)
      READ(20,283) STEP0,PUNCH
283   FORMAT(2F10.0)
      READ(20,284) UNIT,UNAR
284   FORMAT(2F10.0)
      READ(20,285) SILL
285   FORMAT(F10.3)
C
      open(21,file=bathym,status='old')
      open(22,file=nodes,status='old')
      open(23,file=tidein,status='old')
      open(24,file=datout,status='unknown')
      open(16,file=temp,status='unknown')
```

```
c
C..READ GLOBAL COORDINATE OF EACH NODE
      WRITE(16,298)
298   FORMAT(IH,' NODE NUMBER',5X,'X',9X,'Y',9X,'DEPTH',/)
      DO 205 I=1,NN
      READ(21,300)MEXT(I),MB(I),X(I),Y(I),H(I)
300   FORMAT(I4,I1,2F10.5,F3.1)
205   MINT(MEXT(I)=I
      WRITE(16,301)MEXT(I),I,MB(I),X(I),Y(I),H(I),I=1,NN
301   FORMAT(2(3I5,3F10.3))
C
C..READ ELEMENT DATA
C
      READ(22,302)N1(I),N2(I),N3(I),I=1,NM)
302   FORMAT(3I3)
      WRITE(16,303)
303   FORMAT(1H ,'THE ELEMENT CONNECTIONS ',/)
      WRITE(16,304) (I,N1(I),N2(I),N3(I),I=1,NM)
304   FORMAT(5(I7,2X,3I4)
C
C..READ OPEN AND LAND BOUNDARY
C
      READ(22,306)MHNO,(MH(I),I=1,MHNO)
      READ(22,306)MLNO,(MLE(I),I=1,MLNO)
306   FORMAT(16I5)
      WRITE(16,307)MHNO
307   FORMAT(1H,'THERE ARE',I5,'NODES ON THE OPEN BOUNDARY')
      WRITE(16,309) (MH(I),I=1,MHNO)
      WRITE(16,308)MLNO
308   FORMAT(1H,'THERE ARE',I5,'NODES ON THE OPEN BOUNDARY')
      WRITE(16,309) (MLE(I),I=1,MLNO)
309   FORMAT(3X,15I5)
      READ(22,310)(ANG(I),I=1,MLNO)
310   FORMAT(8F10.4)
      WRITE(16,311)
311   FORMAT(1H ,' THE OUTWARD DIRECTIONS ',/)
      WRITE(16,312)(MLE(I),ANG(I),I=1,MLNO)
312   FORMAT(5(I8,F10.4))
      STEBC=1.e10
      NBAND=0
      CRHO=0.00114
      G=32.174
      CDRAG=0.0025
```

```
      F=3.141592/21600.0*SIN(ATTD/180.0*3.141592)
      NWN=2*NW-1
      NN1=NN*2
      NW1=NW*2+1
      CCX=CDRAG*CRHO*ABS(WX)*WX
      CCY=CDRAG*CRHO*ABS(WY)*WY
      C1=1.0/3.0
      C2=1.0/3.0
      C3=1.0/3.0
      C4=1.0/12.0
      C5=1.0/12.0
      C6=1.0/12.0
      C7=1.0/6.0
      C8=1.0/6.0
      C9=1.0/6.0
      DO 201 I=1,NN
      X(I)=X(I)*UNIT
201   Y(I)=Y(I)*UNIT
      DO 202 I=1,MLNO
202   ANG(I)=ANG(I)*DATAN(6.1001)/45.0
      DO 204 I=1,NN1
      DO 204 J=1,NWN1
204   B1(I,J)=0.0
      DO 203 I=1,NN
      DO 203 J=1,NWN
203   B2(I,J)=0.0
C
C
      DO 10 I=1,NM
      N1(I)=MINT(N1(I))
      N2(I)=MINT(N2(I))
      N3(I)=MINT(N3(I))
      L1=IABS(N1(I)-N2(I))+1
      L2=IABS(N2(I)-N3(I))+1
      L3=IABS(N3(I)-N1(I))+1
      IF(NBAND.LT.L1) NBAND=L1
      IF(NBAND.LT.L2) NBAND=L2
      IF(NBAND.LT.L3) NBAND=L3
C
C..TRANSFER TO LOCAL COORDINATE
C
      XP1=(2.0*X(N1(I))-X(N2(I))-X(N3(I)))/3.0
      XP2=(2.0*X(N2(I))-X(N1(I))-X(N3(I)))/3.0
```

```
      XP3=(2.0*X(N3(I))-X(N1(I))-X(N2(I)))/3.0
      YP1=(2.0*Y(N1(I))-Y(N2(I))-Y(N3(I)))/3.0
      YP2=(2.0*Y(N2(I))-Y(N1(I))-Y(N3(I)))/3.0
      YP3=(2.0*Y(N3(I))-Y(N1(I))-Y(N2(I)))/3.0
C
C..CALCULATE ELEMENT AREA
C
      AREA=0.5*(XP2*YP3+XP1*YP2+XP3*YP1-XP2*YP1-XP3*YP2-XP1*YP3)
C
C..CALCULATE MANNING FACTOR
C
      CMAN(I)=0.03
C
C..CALCULATE COEFFIECIENTS OF SHAPE FUNCTION
C
      A(1,I)=(XP2*YP3-XP3*YP2)/2.0*AREA(I))
      A(2,I)=(YP2-YP3)/2.0*AREA(I))
      A(3,I)=(XP3-XP2)/2.0*AREA(I))
      A(4,I)=(XP3*YP1-XP1*YP3)/2.0*AREA(I))
      A(5,I)=(YP3-YP1)/2.0*AREA(I))
      A(6,I)=(XP1-XP3)/2.0*AREA(I))
      A(7,I)=(XP1*YP2-XP2*YP1)/2.0*AREA(I))
      A(8,I)=(YP1-YP2)/2.0*AREA(I))
      A(9,I)=(XP2-XP1)/2.0*AREA(I))
C
C
      AREA(I)=AREA(I)*UNAR
      DIS1=SQRT((X(N1(I))-X(N2(I)))**2+(Y(N1(I))-Y(N2(I)))**2)
      DIS2=SQRT((X(N2(I))-X(N3(I)))**2+(Y(N2(I))-Y(N3(I)))**2)
      DIS3=SQRT((X(N3(I))-X(N1(I)))**2+(Y(N3(I))-Y(N1(I)))**2)
      DH1A=DIS1/SQRT(G*H(N1(I)))
      DH1B=DIS1/SQRT(G*H(N2(I)))
      DH2A=DIS2/SQRT(G*H(N2(I)))
      DH2B=DIS2/SQRT(G*H(N3(I)))
      DH3A=DIS3/SQRT(G*H(N3(I)))
      DH3B=DIS3/SQRT(G*H(N1(I)))
      IF(STEBC.GT.DH1A)  STEBC=DH1A
      IF(STEBC.GT.DH1B)  STEBC=DH1B
      IF(STEBC.GT.DH2A)  STEBC=DH2A
      IF(STEBC.GT.DH2B)  STEBC=DH2B
      IF(STEBC.GT.DH3A)  STEBC=DH3A
      IF(STEBC.GT.DH3B)  STEBC=DH3B
C
```

```
C..FORM THE GLOBAL MATRIX
C
      II=N1(I)
      JJ=N2(I)
      KK=N3(I)
      II1=NW
      JJ1=JJ-(II-NW)
      KK1=KK-(II-NW)
      B2(II,II1)=B2(II,II1)+C7*AREA(I)
      B2(II,JJ1)=B2(II,JJ1)+C4*AREA(I)
      B2(II,KK1)=B2(II,KK1)+C6*AREA(I)
      II1=II-(JJ-NW)
      JJ1=NW
      KK1=KK-(JJ-NW)
      B2(JJ,II1)=B2(JJ,II1)+C4*AREA(I)
      B2(JJ,JJ1)=B2(JJ,JJ1)+C8*AREA(I)
      B2(JJ,KK1)=B2(JJ,KK1)+C5*AREA(I)
      II1=II-(KK-NW)
      JJ1=JJ-(KK-NW
      KK1=KK-NW
      B2(KK,II1)=B2(KK,II1)+C6*AREA(I)
      B2(KK,JJ1)=B2(KK,JJ1)+C5*AREA(I)
      B2(KK,KK1)=B2(KK,KK1)+C9*AREA(I)
10    CONTINUE
      WRITE(16,320) NBAND,STEBC,UNAR
320   FORMAT(1H0,' THE BANDWIDTH =  ',I5,'THE SMALLEST L/SQRT(GH) ='
     &   ,F10.,//,1H,' ELEMENT AREA (UNIT',F10.2,' SQRT FEET)')
      WRITE(16,321)(I,AREA(I),I=1,MN)
321   FORMAT(5(I6,F12.4))
      DO 12 I=1,NN
      DO 12 J=1,NWN
      B1(2*I-1,2*J)=B2(I,J)
12    B1(2*I,2*J)=B1(2*I-1,2*J)
C
C..ROTATE COORDINATE ON THE BOUNDARY
      LB1=NWN-1
      LB2=(NN1-NW1+2)/2
      DO 13 NI=1,MLNO
      I=MINT(MLE(NI))
      NMB=MB(I)
      GO TO (14,13,14,13),NMB
14    CALL ROTB1(B1,ANG(NI),NN1,NW1,NWN1,NW,I,LB1,LB2)
13    CONTINUE
```

```
      DO 23 NI=1,MLNO
      I=MINT(MLE(NI))
      NMB=MB(I)
      GO TO (15,23,15,21),NMB
15    DO 16 J=1,NWN1
16    B1(2*I-1,J)=0.0
      B1(2*I-1,NW1)=1.0
      GO TO 23
21    DO 22 J=1,NWN1
      B1(2*I-1,J)=0.0
22    B1(2*I,J)=0.0
      B1(2*I,NW1)=1.0
      B1(2*I-1,NW1)=1.0
23    CONTINUE
C
C..L U DECOMPOSE MATRIX
C
      CALL MATRIX(B1,NN1,NW1,NWN1)
      DO 17 NI=1,MHNO
      I=MINT(MH(NI))
      DO 18 J=1,NWN
18    B2(I,J)=0.0
      B2(I,NW)=1.0
17    CONTINUE
      CALL MATRIX(B2,NN,NW,NWN)
C
C
C..SET INITIAL VALUES
C
      READ(23,318)(I,TIDE(I),U(I),V(I),I=1,NN)
      DO 9 I=1,NN
      if(TIDE(I).GT.sill)THEN
      ETA(I)=TIDE(I)
      ELSE
      ETA(I)=SILL+0.00001
      ENDIF
      Q(2*I-1)=U(I)*(ETA(I)+H(I))
9     Q(2*I)=V(I)*ETA(I)+H(I))
C
      KL=0
      ML=0
      MLL=1
C
```

```
C..START SEMI-IMPLICIT METHOD
C
      IF(KL.EQ.ML*STEP0) GO TO 27
      GO TO 28
27    ML=ML+1
      TIM=T/60.0
      WRITE(16,315)TIM
315   FORMAT(/,' TIME='F10.2)
      WRITE(16,316)
316   FORMAT(2X,'ELEMENT  TIDE     U VELOCITY  V  VELOCITY')
      WRITE(16,317)(I,MEXT(I),ETA(I),U(I),V(I),I=1,NN)
317   FORMAT(3(2I4,3F10.3))
28    IF(KL.EQ.MLL*PUNCH) GO TO 34
      GO TO 36
34    MLL=MLL+1
      WRITE(24,318)(I,ETA(I),U(I),V(I),I=1,NN)
318   FORMAT(I5,3F1.4,I5,3F11.4)
36    CALL WH(T,HH,WW,HA)
      DO 38 NI=1,MHNO
      I=MINT(MH(NI))
      ETA(I)=HH
38    CONTINUE
39    DO 40 I=1,NN
      R1(2*I-1)=0.0
40    R1(2*i)=0.0
      DO 42 I=1,NM
      UU=Q(2*N1(I)-1)**2+Q(N*N2(I)-1)**2+Q(2*N3(I)-1)**2
      VV=Q(2*N1(I))**2+Q(N*N2(I))**2+Q(2*N3(I))**2
      AVEG=((UU+VV)/3.0)**0.5
      HE1=H(N1(I))+ETA(N1(I))
      HE2=H(N2(I))+ETA(N2(I))
      HE3=H(N3(I))+ETA(N3(I))
      AVEG=((HE1+HE2+HE3)/3.0)**2
      CHEZ(I)=1.49/CMAN(I)*((HE1+HE2+HE3)/3.0)**(1.0/6.0)
      C746E=C7*ETA(N1(I))+C4*H(N2(I))+C6*ETA(N3(I))
      C485E=C4*ETA(N1(I))+C8*H(N2(I))+C5*ETA(N3(I))
      C659E=C6*ETA(N1(I))+C5*H(N2(I))+C9*ETA(N3(I))
      C746H=C7*H(N1(I))+C4*H(N2(I))+C6*H(N3(I))
      C485H=C4*H(N1(I))+C8*H(N2(I))+C5*H(N3(I))
      C659H=C6*H(N1(I))+C5*H(N2(I))+C9*H(N3(I))
      A258E=A(2,I)*ETA(N1(I))+A(5,1)*ETA(N2(I))+A(8,1)*ETA(N3(I))
      A369E=A(3,I)*ETA(N1(I))+A(6,1)*ETA(N2(I))+A(9,1)*ETA(N3(I))
      PREX1=G*(C746H+C746E)*A258E
```

```
 PREX2=G*(C485H+C485E)*A258E
 PREX3=G*(C659H+C659E)*A258E
 PREY1=G*(C746H+C746E)*A369E
 PREY2=G*(C485H+C485E)*A369E
 PREY3=G*(C659H+C659E)*A369E
 C746U=C7*Q(2*N1(I)-1)+C4*Q(2*N2(I)-1)+C6*Q(2*N3(I)-1)
 C485U=C4*Q(2*N1(I)-1)+C8*Q(2*N2(I)-1)+C5*Q(2*N3(I)-1)
 C659U=C6*Q(2*N1(I)-1)+C5*Q(2*N2(I)-1)+C9*Q(2*N3(I)-1)
 C746V=C7*Q(2*N1(I))+C4*Q(2*N2(I))+C6*Q(2*N3(I))
 C485V=C4*Q(2*N1(I))+C8*Q(2*N2(I))+C5*Q(2*N3(I))
 C659V=C6*Q(2*N1(I))+C5*Q(2*N2(I))+C9*Q(2*N3(I))
 QXX1=Q(2*N1(I)-1)**2/HE1
 QXX2=Q(2*N2(I)-1)**2/HE2
 QXX3=Q(2*N3(I)-1)**2/HE3
 QYY1=Q(2*N1(I))**2/HE1
 QYY2=Q(2*N2(I))**2/HE2
 QYY3=Q(2*N3(I))**2/HE3
 QXY1=Q(2*N1(I)-1)*Q(2*N1(I))/HE1
 QXY2=Q(2*N2(I)-1)*Q(2*N2(I))/HE2
 QXY3=Q(2*N3(I)-1)*Q(2*N3(I))/HE3
 CONU=(A(2,I)*QXX1+A(5,I)*QXX2+A(8,1)*QXX3+
&A(3,I)*QXY1+A(6,I)*QXY2+A(9,I)*QXY3)/3.0
 CONV=(A(2,I)*QXY1+A(5,I)*QXY2+A(8,1)*QXY3+
&A(3,I)*QYY1+A(6,I)*QYY2+A(9,I)*QYY3)/3.0
 GACA=G*AVEG/(CHEZ(I)**2)/AVEGH
 FRU1=GACA*C746U
 FRV1=GACA*C746V
 FRU2=GACA*C485U
 FRV2=GACA*C485V
 FRU3=GACA*C659U
 FRV3=GACA*C659V
 UK1= (CONU+PREX1-F*C746V-CCX+FRU1)*AREA(I)
 UK2= (CONU+PREX2-F*C485V-CCX+FRU2)*AREA(I)
 UK3= (CONU+PREX3-F*C659V-CCX+FRU3)*AREA(I)
 VK1= (CONV+PREY1-F*C746V-CCX+FRU1)*AREA(I)
 VK2= (CONV+PREY2-F*C485V-CCX+FRU2)*AREA(I)
 VK3= (CONV+PREY3-F*C659V-CCX+FRU3)*AREA(I)
 II=N1(I)
 JJ=N2(I)
 KK=N3(I)
 R1(II*2-1)=R1(II*2-1)-UK1
 R1(JJ*2-1)=R1(JJ*2-1)-UK2
 R1(KK*2-1)=R1(KK*2-1)-UK3
```

```
      R1(II*2)=R1(II*2)-VK1
      R1(JJ*2)=R1(JJ*2)-VK2
42    R1(KK*2)=R1(KK*2)-VK3
C
      T=T+DT/2.0
      DO 50 NI=1,MLNO
      I=MINT(MLE(NI))
      NMB=MB(I)
      GO TO (49,50,49,52),NMB
52    R1(2*I-1)=0.0
      R1(2*I)=0.0
      GO TO 50
49    CALL ROTUV(R1(2*I-1),R1(2*I),ANG(NI))
      R1(2*I-1)=0.0
50    CONTINUE
      CALL SOLVE(B1,R1,UVP,NN1,NW1,NWN1)
      DO 53 NI=1,MLNO
      I=MINT(MLE(NI))
      NMB=MB(I)
      GO TO (54,53,54,53),NMB
54    CALL ROTUV(UVP(2*I-1)+DT*UVP(2*I),-ANG(NI))
53    CONTINUE
      DO 56 I=1,NN
      Q(2*I-1)=Q(2*I-1)+DT*UVP(2*I-1)
56    Q(2*I)=Q(2*I)+DT*UVP(2*I)
      DO 62 I=1,NN
      HE=H(I)+ETA(I)
      U(I)=Q(2*I-1)/HE
      V(I)=Q(2*I)/HE
62    R2(I)=0.0
      DO 64 I=1,NM
      EK=(A(2,1)*Q(2*(N1(I))-1)+A(5,1)*Q(2*(N2(I))-1)+
     &    A(8,1)*Q(2*(N3(I))-1)+A(3,1)*Q(2*(N1(I))-1)
     &    A(6,1)*Q(2*(N2(I)))+A(9,1)*Q(2*(N3(I))))/3.0*AREA(I)
      II=N1(I)
      JJ=N2(I)
      KK=N3(I)
      R2(II)=R2(II)-EK
      R2(JJ)=R2(JJ)-EK
64    R2(KK)=R2(KK)-EK
      T=T+DT/2.0
      CALL WHP(T,HP,WW,HA)
      DO 66 NI=1,NHNO
```

```
      I=MINT(MH(NI))
      EP(I)=HP
      R2(I)=EP(I)
66    CONTINUE
      CALL SOLVE(B2,R2,EP,NN,NW,NWN)
      DO 68 I=1,NN
68    ETA(I)=ETA(I)+DT*EP(I)
      KL=KL+1
      IF(T-TLIM) 25,25,70
70    STOP
      END
C
      SUBROUTINE WH(T,HH,WW,HA)
      HH=COS(2.0*3.141592*T/WW)*HA
      RETURN
      END
C
      SUBROUTINE WHP(T,HP,WW,HA)
      HP=-2.0*3.141592/WW*SIN(2.0*3.141592*(T/WW))*HA
      RETURN
      END
C
C..LU DECOMPOSITION
C
      SUBROUTINE MATRIX(A,N,NW,NWN)
      DIMENSION A(N,NWN)
      M=N-1
      DO 30 K=1,M
      I1=K+1
      NW1=NW+K-1
      IF(NW1.le.N) GO TO 20
      NW1=N
20    DO 30 I=I1,NW1
      NI=NW-1+I1
      Y=A(I,NI-1)/A(K,NW)
      A(I,NI-1)=Y
      NW11=NI+NW-2
45    DO 30 J=NI,NW11
      A(I,J)=A(I,J)-Y*A(K,J+I-K)
30    CONTINUE
      RETURN
      END
C
```

```
      SUBROUTINE SOLVE(A,B,X,N,NW,NWN)
      DIMENSION A(N,NWN),B(N),X(N)
      X(1)=B(1)
      K1=NW-1
      DO 60 K=2,N
      SUM=0.0
      NW2=NW-K+1
      IF(K.le.NW) GO TO 54
      NW2=1
54    DO 55 J=NW2,K1
55    SUM=SUM+A(K,J)*X(J-NW+K)
60    X(K)=B(K)-SUM
      X(N)=X(N)/A(N,NW)
      K=N
      NW4=NW+1
62    SUM=0.0
      K=K-1
      NW3=NW+N-K
      IF(NW3.LT.NWN) GO TO 64
      NW3=NWN
64    DO 65 J=NW4,NW3
65    SUM=SUM+A(K,J)*X(J-NW+K)
      X(K)=(X(K)-SUM)/A(K,NW)
      IF(K.EQ.1) GO TO 80
      GO TO 62
80    RETURN
      END
C
C..COORDINATE POSITION
C
      SUBROUTINE ROTB1(B1,ANGLE,NN1,NWN1,NWNI,LB1,LB2)
      DIMENSION B1(NN1,NWN1)
      NWM=NW-1
      IF(I.LE.NW) GO TO 20
      LB=LB1
9     DO 10 J=NW,LB
      II=I-J+NW-1
      CALL ROTSM(B1(2*II-1,2*J+2),B1(2*II-1,2*J+3),B1(2*II,2*J+1),
     &B1(2*II,2*J+2),ANGLE)
      NWJ=4*NW-2*J
      B1(2*I,NWJ-2)=B1(2*II-1,2*J+2)
      B1(2*I,NWJ-3)=B1(2*II-1,2*J+3)
      B1(2*I-1,NWJ-1)=B1(2*II,2*J+1)
```

```
      B1(2*I-1,NWJ-2)=B1(2*II,2*J+2)
10    CONTINUE
      GO TO 30
20    IF(I.EQ.1)GO TO 30
      LB=I+NW-2
      GO TO 9
30    IF(I.GT.LB2 GO TO 60
      LB=1
50    DO 70 J=LB,NWM
      II=I-J+NW
      CALL ROTSM(B1(2*II-1,2*J),B1(2*II-1,2*J+1),B1(2*II,2*J-1),
     &B1(2*II,2*J),ANGLE)
      NWJ=4*NW-2*J
      B1(2*I,NWJ)=B1(2*II-1,2*J)
      B1(2*I,NWJ-1)=B1(2*II-1,2*J)
      B1(2*I-1,NWJ+1)=B1(2*II-1,2*J-1)
      B1(2*I-1,NWJ)=B1(2*II,2*J)
70    CONTINUE
      GO TO 90
60    IF(I.EQ.NN1/2)GO TO 90
      LB=1+(I-LB2)
90    RETURN
      END
C
      SUBROUTINE ROTSM(SM1,SM2,SM3,SM4,ANGLE)
      SM1P=SM1*COS(ANGLE)+SM2*SIN(ANGLE)
      SM2=SM1*SIN(ANGLE)+SM2*COS(ANGLE)
      SM3P=SM3*COS(ANGLE)+SM4*SIN(ANGLE)
      SM4=SM3*SIN(ANGLE)+SM4*COS(ANGLE)
      SM1=SM1P
      SM3=SM3P
      RETURN
      END
C
      SUBROUTINE ROTUV(A,B,ANGLE)
      AP=A*COS(ANGLE)+B*SIN(ANGLE)
      B=A*SIN(ANGLE)+B*COS(ANGLE)
      A=AP
      RETURN
      END
C
```

APPENDIX B

CODE FOR THE TID_DAYS TIDAL FORCING MODEL

Appendix B:
Code for the TID_DAYS Tidal Forcing Model

```
c: Tidal Forcing Model TID_DAYS.FOR
c           Sep 23, 2004
c..only does 1 month at user selected interval TIMER
c..modified STORX and u2 arrays to 50000
c..outputs in feet NGVD
c.....6 minute
C  This program outputs the predicted tidal amplitudes at 1 hr
C  intervals for a 24 hr period beginning at local time 00:00 of
C  January of the year selected in the input file.  The input file
C  must contain the amplitude and phase of the local tidal constituents
C  (up to 37 constituents).
C  Uses Long's code for the prediction of tides using harmonic
C  equations from U.S. Dept of Commerse SP #98 1988
C  by Scott A. Jenkins & Joseph Wasyl
c
C*************************************************************
       character nameref*60,ifile*60
      LOGICAL          MKTABLE
      INTEGER      STARTDATE(3),  BEGIN_DAY(12),  END_DAY(12)
      INTEGER            MO(12)
      DIMENSION          A(37),       AMP(37),    PHASE(37)
      DIMENSION        SPD(37),      AMPA(37),    EPOCH(37)
      DIMENSION     STORX(50000)
      DIMENSION       YODE(37),      YVPU(37)
      DIMENSION      JDAYF(12),     JDAYL(12),    JWKDA(12)
C
C  Tidal constituant speeds in degrees per hour
C
c  Tidal constituents speed, amplitude, phase angle read from data
c  file tideconsts.dat (degrees/hr, feet, degrees)
c
c
      DATA       MO/  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12 /
C
      DATA BEGIN_DAY/  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1 /
C
      DATA   END_DAY/ 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 /
C
      COMMON /AC1/    SPD,     EPOCH,       AMPA
      COMMON /AC2/  STORX,       MO,   BEGIN_DAY,   END_DAY,
```

```
     &           NUMCON,   MKTABLE,        JP,     LPYR,
     &            TIMER,     NOHRS
C
C  PRELIMINARY CONSTANTS
C
      MKTABLE=.TRUE.                 !MAKE A TABLE
      CON=2048./90.                  !UNITS/DEG, DISCRETE COS LOOP
      NUMCON=37                      !# OF TIDAL CONSTITUANTS
c* read from input file      TIMER=1.  !# OF PREDICTIONS/HOUR
      NUM_MONTHS=12                  !# OF MONTHS TO RUN
c
c
      open(25,file='tides1mo.inp',status='old')
c
      read(25,*)IYR
      read(25,*)IMO
      read(25,*)TIMER
      read(25,'(a)')nameref
      close(25)
c
       JP=IMO
c
        write(ifile,1000)nameref(1:LSTGDCHR(nameref))
c
1000    format(a)
c
c.....tidal constituents from file name in input file 'tide_1yr.inp'
c
      open(26,file=ifile,status='old')
c
      do 49 ic=1,numcon
      read(26,*)A(ic),AMP(ic),PHASE(ic)
49    continue
c
      read(26,*)c1
      close(26)
c
       M2=AMP(1)
c
C
C  COMPUTE INDEX OF THE WEEKDAY OF 1 JAN OF COMPUTATION YEAR
C  (1=SUNDAY, 2=MONDAY, ... , 7=SATURDAY)
C
```

```
      IDAY11=1+MOD((IYR-1+INT(FLOAT(IYR-1)/4.)),7)
C
C  CHECK FOR AND DEAL WITH LEAP YEAR
C
      LPYR=0
      NDAYS=365
      IF(MOD(IYR,4).EQ.0) THEN
        LPYR=1
        END_DAY(2)=29
        NDAYS=366
      END IF
C
C  SET STARTING DATE
C
      STARTDATE(1)=IYR              !YEAR
      STARTDATE(2)=1                !MONTH
      STARTDATE(3)=1                !DAY
C
C  SET EQUILIBRIUM ARGUMENTS (?)
C
      CALL EQU(NUMCON,STARTDATE,NDAYS,YODE,YVPU)
C
C  OUTPUT CONTENTS OF YODE AND YVPU (COMMENT OUT IF NOT NEEDED)
C
C     OPEN(11,FILE='EQUOUT.DAT',STATUS='UNKNOWN',FORM='FORMATTED')
C     WRITE(11,'(1X," YODE(J)",7X," YVPU(J)",/)')
C     DO 10 J=1,NUMCON
C       WRITE(11,'(1X,F8.3," ----- ",F8.3)') YODE(J),YVPU(J)
C  10 CONTINUE
C     CLOSE(11)
C
      WRITE(*,'(1X,/,1X,"Calculating tide predictions.",1X,/)')
C
C  CREATE OPERATING ARRAYS:  AMPA, EPOCH, SPD
C
      DO 20 J=1,NUMCON
        AMPA(J)=AMP(J)*YODE(J)
        TEMX=YVPU(J)-PHASE(J)
        IF(TEMX .LT. 0.) TEMX=TEMX+360.
        EPOCH(J)=TEMX*CON
        SPD(J)=A(J)*CON/TIMER
   20 CONTINUE
C
```

```
C  COMPUTE JULIAN START AND END DAYS AND WEEK DAY OF 1ST OF
C  EACH MONTH BASED ON END_DAY ARRAY AND IDAY11
C
      JDAYF(1)=1
      JDAYL(1)=END_DAY(1)
      JWKDA(1)=IDAY11
      DO 25 N=2,12
        JDAYF(N)=JDAYL(N-1)+1
        JDAYL(N)=JDAYL(N-1)+END_DAY(N)
        JWKDA(N)=1+MOD( (JDAYF(N)+IDAY11-2) ,7)
   25 CONTINUE
C
C  CALL ROUTINES TO DO TIDAL PREDICTION CALCULATIONS
C
C************Only calculate month from input file**********
c......      DO 30 JP=1,NUM_MONTHS
        CALL CTIDE
        CALL TIDEOUT(M2,c1,IYR,IMO,IDAY,JDAYF(JP),JDAYL(JP),JWKDA(JP))
c......   30 CONTINUE
C
      WRITE(*,'(1X,/,1X,"Tide calculations complete.")')
C
      END
C
C
C
C
      SUBROUTINE CTIDE
C
C  COMPUTES  TIMER  PREDICTIONS PER HOUR OF THE TIDES AND
C  STORES THE RESULT IN ARRAY STORX.
C  NUMCON =  MAX NUMBER OF TIDAL CONSTITUENTS
C  MKTABLE = LOGICAL FLAG;  WHEN .TRUE. IT MAKES A TABLE
C           OF 8193 COSINES (SAVED IN ARRAY XCOS) THAT
C           REPRESENT 360 DEGREES FOR A TABLE LOOK-UP OF
C           THE TIDAL COSINE FUNCTION
C  TABHR =   HOURS TO THE BEGINNING OF EACH MONTH, OFFSET
C           BY 24.  FIRST 12 ELEMENTS FOR A NORMAL YEAR;
C           SECOND 12 FOR LEAP YEAR.
C
      LOGICAL           MKTABLE
      INTEGER      BEGIN_DAY(12),  END_DAY(12),      MO(12)
      DIMENSION       XCOS(8193),      ARG(37)
```

```
      DIMENSION         SPD(37),    EPOCH(37),    AMPA(37)
      DIMENSION       TABHR(24),  STORX(50000)
      DOUBLE PRECISION        H
C
      COMMON /AC1/          SPD,       EPOCH,       AMPA
      COMMON /AC2/        STORX,          MO,   BEGIN_DAY,
     &                 END_DAY,      NUMCON,     MKTABLE,
     &                      JP,        LPYR,      TIMER,
     &                   NOHRS
C
      DATA TABHR/  -24.,  720., 1392., 2136., 2856., 3600.,
     &           4320., 5064., 5808., 6528., 7272., 7992.,
     &            -24.,  720., 1416., 2160., 2880., 3624.,
     &           4344., 5088., 5832., 6552., 7296., 8016./
C
      IF(MKTABLE) THEN
        H=8.D0*DATAN(1.D0)/8192.D0
        DO 10 I=1,8193
          XCOS(I)=SNGL(DCOS(DFLOAT(I-1)*H))
10        CONTINUE
        MKTABLE=.FALSE.
      END IF
C
C  SET DATES AND TIMES
C
      NOD=END_DAY(JP)-BEGIN_DAY(JP)+1        !# OF DAYS
      NOHRS=(NOD*24)*NINT(TIMER)+1           !# OF SAMPLES
      NOHRS=NOHRS+24*NINT(TIMER)             !ADD 12 HRS AT ENDS
c........NOHRS is the total number of tidal realizations
C
C  FIRST = # OF ESTIMATION POINTS (HOURS*TIMER), COUNTING
C  FROM 0000 1 JAN OF PREDICTION YEAR, OF FIRST PREDICTION
C
      FIRST=(TABHR(MO(JP)+12*LPYR)+24.*BEGIN_DAY(JP))*TIMER
      FIRST=FIRST-12.*TIMER                  !START @ 1200, DAY 0
C
C*****************************************************
C  START MAIN COMPUTATION LOOP                       *
C*****************************************************
C
      DO 30 K=1,NOHRS
C
C  SET NUMCON VALUES OF ARG(J)
```

```
C
      IF(K.EQ.1) THEN
        DO 40 J=1,NUMCON
          ARGU=SPD(J)*FIRST+EPOCH(J)
          ARG(J)=AMOD(ARGU,8192.)
          IF(ARG(J).LT.0.) ARG(J)=ARG(J)+8192.
  40      CONTINUE
      ELSE
        DO 50 J=1,NUMCON
          ARG(J)=ARG(J)+SPD(J)
          IF(ARG(J).GE.8192.) ARG(J)=ARG(J)-8192.
  50      CONTINUE
      END IF
C
C  SUM NUMCON CONSTITUENT CONTRIBUTIONS FOR ONE TIDAL ELEVATION
C
      TIDE=0.
      DO 60 J=1,NUMCON
        NP=INT(ARG(J)+1.5)
        TIDE=TIDE+AMPA(J)*XCOS(NP)
  60    CONTINUE
C
C  SET FINAL OUTPUT
C
      STORX(K)=(TIDE+0.26)c..convert from feet MSL to feet NGVD C
  30 CONTINUE
C
C*******************************************************
C  END OF MAIN COMPUTATION LOOP                          *
C*******************************************************
C
    RETURN
    END
C
C
C
C
    SUBROUTINE TIDEOUT(M2,c1,IYR,IMO,IDAY,JDAYB,JDAYE,JDAYW)
C
C  WRITES TIDAL PREDICTION ARRAY AND COMPUTED TIME TO AN
C  OUTPUT FILE.  JDAYB AND JDAYE ARE BEGINNING AND ENDING
C  JULIAN DAYS OF THE JP'TH MONTH.  JDAYW IS THE WEEKDAY OF
C  THE 1ST OF THE MONTH (1 = SUNDAY, 2 = MONDAY, ... ,
```

```
C  7 = SATURDAY)
C
      LOGICAL            MKTABLE
      CHARACTER*2            CYR
      CHARACTER*3     MONAME(12)
      CHARACTER*10        FNAME
      INTEGER       BEGIN_DAY(12),   END_DAY(12),     MO(12)
      DIMENSION       STORX(50000),  u0(50000)
C
      COMMON /AC2/         STORX,           MO,  BEGIN_DAY,
     &                  END_DAY,       NUMCON,    MKTABLE,
     &                       JP,         LPYR,      TIMER,
     &                    NOHRS
C
      DATA MONAME/ 'Jan','Feb','Mar','Apr','May','Jun',
     &            'Jul','Aug','Sep','Oct','Nov','Dec'/
C
C  CORRECTION MINUTES, HOURS AND DAYS FOR OFFSET START TIME
C
      IMNOFF=0
      IHROFF=0
      IDAOFF=0
C
C  NAME OUTPUT FILE
C
      WRITE(CYR,'(I2)') MOD(IYR,100)
      FNAME=MONAME(JP)//CYR//'.DAT'
      WRITE(*,*) ' Writing to ', FNAME
C
      OPEN(10,FILE=FNAME,STATUS='UNKNOWN',FORM='FORMATTED')
C
c********comment out the bullshit headers*******
c      WRITE(10,'(2X,"Tides for ",A3,", ",I4)') MONAME(JP), IYR
c      WRITE(10,'(2X,"Julian days ",I3," to ",I3)') JDAYB, JDAYE
c      WRITE(10,'(2X,I1," = weekday of ",A3," 1st",/)') JDAYW,
c     &   MONAME(JP)
c      WRITE(10,'(2X," date/time",3X,"m (NGVD)",/)')
C
      PLUSTIME=60./TIMER
      IMN=-PLUSTIME + IMNOFF             !START @ 0 MIN + FIX
      IHR=12 + IHROFF                    !START @ NOON + FIX
      IDA=BEGIN_DAY(JP) - 1 + IDAOFF      !START @ DAY 0 + FIX
C
```

```
c**********only write out 00:00 day1 - 23:00 day 30,31,28,29)*******
      TIMER12=TIMER*12
      NOHRS12T=NOHRS-12*NINT(TIMER)-1
c
      DO 10 I=1,NOHRS
         IMN=IMN+PLUSTIME
         IF(IMN.GE.60) THEN
            IHR=IHR+1
            IMN=IMN-60
         END IF
         IF(IHR.GE.24) THEN
            IDA=IDA+1
            IHR=IHR-24
         END IF
         ITIME=100*IHR+IMN
         if(I.LE.TIMER12)go to 1111
         if(I.GT.NOHRS12T)go to 1111
c
         Im1=I-1
c***************************tide current module****************************
       eldif=(STORX(I)-STORX(Im1))
       u0(I)=(2.04*9.8*ABS(eldif)*M2/c1)**0.5
       if(eldif.LT.0)dir=0
       if(eldif.GE.0)dir=180
c**************************************************************************
c
        chrs=1.0*(I-TIMER12-1)/TIMER         WRITE(10,'(F10.2,F10.4)')
     &      chrs,STORX(I)
1111     continue
   10 CONTINUE
C
      CLOSE(10)
C
      RETURN
      END
C
C
C
C
      SUBROUTINE EQU(NSPED,IDT,LENGTH,FFF,VAU)
C
C  CALCULATE TIDAL EQUILIBRIUM ARGUMENTS (VAU) AND NODE
C  FACTORS (FFF).  DEVELOPERS:  E.E. LONG, B.B. PARKER,
```

```
C  L. HICKMAN AND G. FRENCH.  NOTES:  VAU IS CALCULATED
C  FOR THE BEGINNING OF THE SERIES;  FFF IS ADJUSTED TO
C  THE MIDPOINT OF THE YEAR BEING CALCULATED
C
      CHARACTER*10       LNAME,  LABEL(37)
      DIMENSION      NODAYS(12),    FFF(37), VAU(37),  IDT(3)
      DIMENSION         CXX(30),     OEX(5)
      DOUBLE PRECISION    SPEED,    SPD(37)
C
      COMMON /LOCAT/    TM,  GONL
      COMMON /COSTX/   CXX,   OEX
      COMMON   /FAD/ IPICK
      COMMON   /VEE/   TML,   CON,    U,    Q,   UI
      COMMON  /BOXA/     S,    XL,   PM,   PL,   SL,   PS,
     &                PLM,  SKYN,   VI,    V,   XI,  VPP
      COMMON  /BOXB/    VP,     P,  AUL,  AUM,  CRA,  CQA
      COMMON  /BOXS/    AW,    AI,   AE,  AE1,  ASP
C
      DATA NODAYS/ 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 /
C
      DATA  LABEL/
     &   'M(2)      ' , 'S(2)      ' , 'N(2)      ' , 'K(1)      ' ,
     &   'M(4)      ' , 'O(1)      ' , 'M(6)      ' , 'MK(3)     ' ,
     &   'S(4)      ' , 'MN(4)     ' , 'Nu(2)     ' , 'S(6)      ' ,
     &   'Mu(2)     ' , '2N(2)     ' , 'OO(1)     ' , 'Lambda(2) ' ,
     &   'S(1)      ' , 'M(1)      ' , 'J(1)      ' , 'Mm        ' ,
     &   'Ssa       ' , 'Sa        ' , 'Msf       ' , 'Mf        ' ,
     &   'Rho(1)    ' , 'Q(1)      ' , 'T(2)      ' , 'R(2)      ' ,
     &   '2Q(1)     ' , 'P(1)      ' , '2SM(2)    ' , 'M(3)      ' ,
     &   'L(2)      ' , '2MK(3)    ' , 'K(2)      ' , 'M(8)      ' ,
     &   'MS(4)     ' /
C
C  SET SOME VARIABLES:
C    NSPED = NUMBER OF CONSTITUENTS TO BE CALCULATED
C    MONTH = MONTH OF FIRST DATA POINT
C     IDAY = DAY OF FIRST DATA POINT
C     IYER = YEAR OF FIRST DATA POINT
C   LENGTH = LENGTH (IN DAYS) OF SERIES TO BE GENERATED
C
      IYER=IDT(1)
      DAYB=0.0
      GRBS=0.0
C
```

```
C  DEAL WITH LEAP YEAR FEBRUARY, DAYS/YEAR AND MIDDLE DAY/TIME
C
      IF(MOD(IYER,4) .EQ. 0) THEN
        NODAYS(2)=29
        NDAY=366
        DAYM=184.0
        GRMS=0.0
      ELSE
        NODAYS(2)=28
        NDAY=365
        DAYM=183.0
        GRMS=12.0
      END IF
C
      XYER=FLOAT(IYER)
C
C  CALL ROUTINE TO COMPUTE BASIC ASTRONOMICAL CONSTANTS
C
      CALL ASTRO(XYER,DAYB,DAYM,GRBS,GRMS)
C
C  MORE CONSTANTS FOR COMMON BLOCKS
C
      TM=0.0
      GONL=0.0
      TML=0.0
      JUDAY=IDT(3)
C
C  LOOK UP CONSTITUENT PARAMETERS BY NAME MATCHING
C
      DO 10 J=1,NSPED
        LNAME=LABEL(J)
        SPEED=0.D0
        CALL NAME(SPEED,LNAME,ISUB,INUM,2)
        SPD(J)=SPEED
        CALL VANDUF(SPEED,E,F,1)
        FFF(J)=F
        VAU(J)=E
   10 CONTINUE
C
C  ROUND NODE TO 3 DECIMAL PLACES, VO+U TO 1 DECIMAL PLACE
C
      DO 30 J=1,NSPED
        FFF(J)=ANINT(FFF(J)*1000.)*0.001
```

```
        VAU(J)=ANINT(VAU(J)*10.)*0.1
   30 CONTINUE
C
      RETURN
      END
C
C
C
C
      SUBROUTINE ASTRO(XYER,DAYB,DAYM,GRBS,GRMS)
C
C  COMPUTES ASTRONOMICAL CONSTANTS FOR THE YEAR BEGINNING
C  DAY (DAYB) AND HOUR (GRBS) AND FOR THE YEAR MIDDLE DAY
C  (DAYM) AND HOUR (GRMS)
C
      DIMENSION  CXX(30), OEX(5)
C
      COMMON /LOCAT/   TM,  GONL
      COMMON /COSTX/  CXX,   OEX
      COMMON   /VEE/  TML,   CON,    U,    Q,    UI
      COMMON  /BOXA/    S,    XL,   PM,   PL,    SL,    PS,   PLM,
     &              SKYN,    VI,    V,   XI,   VPP
      COMMON  /BOXB/   VP,     P,  AUL,  AUM,   CRA,   CQA
      COMMON  /BOXS/   AW,    AI,   AE,  AE1,   ASP
      COMMON /BOXXS/  VIB,    VB,  XIB,  VPB,  VPPB,  CXSB,  CXPB,
     &              CXHB, CXP1B
C
      PINV  = 57.29578                !degrees/radian
C
C  ORBIT SETS LUNAR AND SOLAR MEAN LATITUDES FOR THE BEGINNING OF
C  THE CENTURY CONTAINING THE TIDE YEAR XYER (FROM TABLE 1 IN
C  S.P. 98)
C
      CALL ORBIT(XCEN,XSX,XPX,XHX,XP1X,XNX,OEX,T,XYER,5)
C
C  FROM TABLE 1 IN S.P. 98:
C
C   XW = OBLIQUITY OF THE ECLIPTIC = MAX DECLINATION OF THE SUN
C       (w) (DEGREES)
C   XI = INCLINATION OF THE MOON'S ORBIT TO PLANE OF THE ECLIPTIC
C       (i) (DEGREES)
C   AW = XW IN RADIANS
C   AI = XI IN RADIANS
```

```
C   AE = ECCENTRICITY OF MOON'S ORBIT (e)
C  AE1 = ECCENTRICITY OF EARTH'S ORBIT (e1)
C  ASP = SOLAR FACTOR (S')
C
      XW=23.4522944+T*(-0.0130125+T*(-0.00000164+T*(0.000000503)))
      XI  =  5.14537628
      AW  =  0.0174533*XW                              !w
      AI  =  0.0174533*XI                              !I
      AE  =  0.0548997                                 !e
      AE1 =  0.01675104+T*(-0.0000418+T*(-0.000000126))  !e1
      ASP =  0.46022931                                !S'
C
      DO 30 NOE=1,30
        CXX(NOE) = 0.0
   30 CONTINUE
C
C  This next section must have something to do with the 1/4 day per
C  year difference between the Common Year and the Julian Year
C
      IF(DAYB.GT.0.0) DAYB = DAYB-1.       !reduce beginning day by 1
      IF(DAYM.GT.0.0) DAYM = DAYM-1.       !reduce middle day by 1
      AMIT  = 0.0                          !initialize correction
      AMI   = XYER-XCEN                    !years into current century
      CPLEX = XCEN/400.+0.0001             !century/400 + epsilon
      DICF  = CPLEX-AINT(CPLEX)            !remainder after century/400
      IF(AMI .EQ. 0.) GO TO 40             !skip if exact century year
      XCET  = XCEN+1.                      !century year + 1
      CDIF  = XYER-XCET                    !years into century - 1
      DOBY  = CDIF/4.+0.0001               !(yrs in cent - 1)/4 + eps
      AMIT  = AINT(DOBY)                   !truncate (yrs in cent -1)/4
      IF(DICF .LT. 0.001) AMIT = AMIT+1.0  !add 1 if 1600's or 2000's
   40 CONTINUE
C
      DBMT  = DAYB+AMIT                    !fix for 1/4 day/(year in cent)
      DMMT  = DAYM+AMIT                    !        ditto
C
C  FROM TABLE 1, S.P. 98, BOX 4, CXX(1-8) ARE RATES OF CHANGE OF
C  MEAN LATITUDE OF:
C
C    CXX(1) = MOON TO BEGINNING DAY AND BEGINNING HOUR
C    CXX(2) = LUNAR PERIGEE TO BEGINNING DAY AND HOUR
C    CXX(3) = SUN TO BEGINNING DAY AND HOUR
C    CXX(4) = SOLAR PERIGEE TO BEGINNING DAY AND HOUR
```

```
C    CXX(5) = LUNAR PERIGEE TO MIDDLE DAY AND HOUR
C    CXX(6) = MOON'S NODE TO MIDDLE DAY AND HOUR
C    CXX(7) = LUNAR PERIGEE TO MIDDLE DAY AND BEGINNING HOUR
C    CXX(8) = MOON'S NODE TO BEGINNING DAY AND HOUR
C
      CXX(1)=XSX+  129.384820*AMI+  13.1763968*DBMT+0.549016532*GRBS
      CXX(2)=XPX+  40.6624658*AMI+ 0.111404016*DBMT+0.004641834*GRBS
      CXX(3)=XHX- 0.238724988*AMI+ 0.985647329*DBMT+0.041068639*GRBS
      CXX(4)=XP1X+ 0.01717836*AMI+ 0.000047064*DBMT+0.000001961*GRBS
      CXX(5)=XPX+  40.6624658*AMI+ 0.111404016*DMMT+0.004641834*GRMS
      CXX(6)=XNX-  19.3281858*AMI- 0.052953934*DMMT-0.002206414*GRMS
      CXX(7)=XPX+  40.6624658*AMI+ 0.111404016*DMMT+0.004641834*GRBS
      CXX(8)=XNX-19.328185764*AMI-0.0529539336*DBMT-0.002206414*GRBS
C
C  Round CXX(1-8) to nearest 0.01 deg and, if negative, increment
C  by 360 deg (essentially modulo 360 arithmetic)
C
      DO 50 J=1,8
        ZAT = CXX(J)/360.                 !# of loops + fraction
        IF(ZAT .LT. 0.) THEN              !if negative angle
          CXX(J) = ((ZAT-AINT(ZAT))+1.)*360.     !360 + fraction*360
        ELSE                              !if positive angle
          CXX(J) = (ZAT-AINT(ZAT))*360.          !fraction*360
        END IF
        CXX(J) = FLOAT(IFIX(CXX(J)*100.+0.5))*0.01  !nearest 0.01 deg
   50 CONTINUE
C
C  Use latitude of moon's node (N) at beginning day and hour
C
      ANG = CXX(8)
C
C  Set astronomical constants at beginning day and hour
C
      CALL TABLE6(VIB,VB,XIB,VPB,VPPB,XX,XX,XX,XX,XX,ANG,ANB,ATB)
C
      CXX(26) = VIB      !I at beginning day and hour
      CXX(27) = VB       !nu at         "
      CXX(28) = XIB      !zi at         "
      CXX(29) = VPB      !nu' at        "
      CXX(30) = VPPB     !2*nu" at      "
C
      CXSB  = CXX(1)     !s at beginning day and hour
      CXPB  = CXX(2)     !p          "
```

```
      CXHB  = CXX(3)      !h            "
      CXP1B = CXX(4)      !p1           "
C
C  Use latitude of moon's node (N) at middle day and hour
C
      ANG   = CXX(6)
C
C  Set astronomical constants at middle day and hour
C
      CALL TABLE6(VI,V,XI,VP,VPP,CIG,CVX,CEX,PVC,PVCP,ANG,AN,AT)
C
      CXX(9)  = VI         !I at middle day and hour
      CXX(10) = V          !nu at        "
      CXX(11) = XI         !zi at        "
      CXX(12) = VP         !nu' at       "
      CXX(13) = VPP        !2*nu" at     "
C
C  Round CXX(9-13) to nearest 0.01 deg
C
      DO 60 J=9,13
        CXX(J) = FLOAT(IFIX(CXX(J)*100.+0.5))*0.01
   60 CONTINUE
C
C  Define P = p - zi following Eq. 191, para. 122.
C
      PGX = FLOAT(IFIX((CXX(5)-CXX(11))*100.+0.5))*0.01  !define P to 0.01 deg
      ZAT = PGX/360.                                 !modulo 360 math
      IF(ZAT .LT. 0.) THEN
        PGX = ((ZAT-AINT(ZAT))+1.)*360.              !360+fraction*360
      ELSE
        PGX = (ZAT-AINT(ZAT))*360.                   !fraction*360
      END IF
      XPG = PGX*0.0174533                            !P in radians
      CXX(14) = PGX
C
C  For argument of constituent L2, compute R of Eq. 214, para. 129 (which
C  info. is also tabulated in Table 8 of S.P. 98):
C
C              RAXE = sin(2P)
C              RAXN = -cos(2P) + 1/6 [cot(I/2)]**2
C                 R = arctan(RAXE/RAXN)
C
      RAXE    = SIN(2.*XPG)
```

```
      RAXN    = -COS(2.*XPG)+1./(6.*(TAN(0.5*AT)**2))
      RXX     = 0.
      IF(RAXE .EQ. 0. .OR. RAXN .EQ. 0.) GO TO 70
      RAX     = RAXE/RAXN                          !tan(R)
      IF(RAX .GT. 3450.) GO TO 70
      RXX     = ATAN(RAX)*PINV                     !R in degrees
      CXX(22) = RXX
   70 CONTINUE
C
C  For amplitude of constituent L2, compute 1/Ra with Eq. 213 of
C  para. 129 [for which Table 7 lists log(Ra) ]:
C
C    1/Ra = Sqrt[ 1 - 12 tan^2(I/2) cos(2P) + 36 tan^4(I/2) ]
C
      CRA = SQRT(    1.
     &             -12. * (  TAN(0.5*AT)**2  ) * COS(2.*XPG)
     &             +36. * (  TAN(0.5*AT)**4  )    )
C
C  Find constant terms in cosine argument of Eq. 212, para. 129.
C
      UM2     = 2.*(CXX(11)-CXX(10))         !2(zi-nu)
      CXX(21) = UM2                          !2(zi-nu)
      CXX(24) = CRA                          !1/Ra
      U12     = UM2-RXX                      !2(zi-nu)-R
      U12     = U12+180.                     !2(zi-nu)-R+180
      CXX(15) = U12
C
C  Compute Q of Eq. 202, para. 123:
C
C              Q = arctan[(5 cos I - 1)*tan P / (7 cos I + 1)]
C
      ZES     = (5.*COS(AW)-1.) * SIN(XPG)
      ZEC     = (7.*COS(AW)+1.) * COS(XPG)
      CALL FITAN(ZES,ZEC,QXX,SPXX,2)
      CXX(23) = QXX                          !Q in proper quadrant
C
C  For constant terms in cosine argument of Eq. 201, para. 123.
C  Note:  sign on 90 is opposite to that given in S.P. 98
C
      CRAV    = 0.5*UM2+QXX+90.              !zi-nu+Q+90
      CXX(16) = CRAV
C
C  For 1/Qa of Eq. 195, para. 122 for M1 tide:
```

```
C  Note:  changed AW to AT to be consistent with S.P. 98!!!
C
C  -> OLD TERM:    CQA = SQRT(   0.25
C    &             +1.50 * COS(2.*XPG) * COS(AW) / (COS(0.5*AW)**2)
C    &             +2.25 * (COS(AW)**2) / (COS(0.5*AW)**4)    )
C
      CQA = SQRT(   0.25
     &             +1.50 * COS(2.*XPG) * COS(AT) / (COS(0.5*AT)**2)
     &             +2.25 * (COS(AT)**2) / (COS(0.5*AT)**4)    )
C
      CXX(25) = CQA
C
C  Round CXX(14-23) to nearest 0.01 deg
C
      DO 80 J=14,23
        CXX(J) = FLOAT(  IFIX( CXX(J)*100.+0.5 )  )*0.01
   80 CONTINUE
C
C  Give names to some of the array elements
C
      PM   = CXX(1)         !s (beginning day)
      PL   = CXX(2)         !p      "
      SL   = CXX(3)         !h      "
      PS   = CXX(4)         !p1     "
      PLM  = CXX(5)         !p (middle day)
      SKYN = CXX(6)         !N      "
      VI   = CXX(9)         !I      "
      V    = CXX(10)        !nu     "
      XI   = CXX(11)        !zi     "
      VP   = CXX(12)        !nu'    "
      VPP  = CXX(13)        !2*nu"  "
      P    = CXX(14)        !P      "
      AUL  = CXX(15)        !2zi-2nu-R+180  (middle)
      AUM  = CXX(16)        !zi-nu+Q+90       "
      CRA  = CXX(24)        !1/Ra
      CQA  = CXX(25)        !1/Qa
C
      U    =  V*0.0174533    !nu in radians
      Q    =  P*0.0174533    !P  in radians
      UI   = VI*0.0174533    !I  in radians
C
      RETURN
      END
```

```
C
C
C
C
      SUBROUTINE FITAN(AUS,AUC,RTA,SPDX,JMAP)
C*******************************************************
C  THIS APPEARS TO BE A FORM OF THE FUNCTION ATAN2, BUT   *
C  WITH SOME ADDITIONAL STUFF.                          *
C*******************************************************
      IF(AUC .EQ. 0.) THEN
        IF(AUS .LT. 0.) THEN
          RTA = 270.
        ELSE IF(AUS .EQ. 0.) THEN
          RTA = 0.
        ELSE
          RTA = 90.
        END IF
      ELSE
        RTA = 57.2957795*ATAN(AUS/AUC)
        IF(JMAP .EQ. 2) THEN
          IF(AUS .LE. 0.) THEN
            IF(AUC .LT. 0.) THEN
              RTA = RTA+180.
            ELSE IF(AUC .EQ. 0.) THEN
              RTA = 0.
            ELSE
              RTA = RTA+360.
            END IF
          ELSE
            IF(AUC .LT. 0.) THEN
              RTA = RTA+180.
            ELSE IF(AUC .EQ. 0.) THEN
              RTA = 90.
            END IF
          END IF
        END IF
      END IF
C
      SPDX = 0.
C
      RETURN
      END
C
```

```
C
C
C
      SUBROUTINE ORBIT(XCEN,XSX,XPX,XHX,XP1X,XNX,OEX,T,
     &   XYER,NNN)
C
C  Computes fundamental astronomical variables from Table 1 in
C  S.P. 98 (1988 ed.)
C
C    S = rate of change of mean longitude of the moon per solar day
C    P = rate of change of mean longitude of lunar perigee/solar day
C   XH = rate of change of mean longitude of the sun per solar day
C   P1 = rate of change of mean longitude of solar perigee/solar day
C   XN = rate of change of mean longitude of moon's node/solar day
C    T = number of Julian centuries (36525 days) reckoned from
C        Greenwich mean noon, 31 December 1899
C   YR = number of days and half days to correct astronomical
C        constants to 0000 hrs, 1 January of proper century, starting
C        with the year 1600 (correction is 0.5 days for 20th century,
C        i.e., noon on 31 December 1899 to 0000 hrs, 1 January 1900)
C  GAT = the century (XCEN) in which the year XYER resides
C  OEX = array of (NNN =) 5 elements, redundantly returned with the
C        mean latitudes listed below as of 0000 hrs, 1 January of
C        the proper century:
C
C         XSX = OEX(1) = mean longitude of moon (s)
C         XPX = OEX(2) = mean longitude of lunar perigee (p)
C         XHX = OEX(3) = mean longitude of sun (h)
C        XP1X = OEX(4) = mean longitude of solar perigee (p1)
C         XNX = OEX(5) = mean longitude of moon's node (N)
C
      DIMENSION OEX(NNN)
C
      S    = 13.1763968
      P    = 0.1114040
      XH   = 0.9856473
      P1   = 0.0000471
      XN   = -0.0529539
      XCAN = XYER*0.01+0.001
      XCEN = AINT(XCAN)*100.
      T    = -3.0
      YR   = 2.5
      GAT  = 1600.
```

```
C
      DO 10 JK=1,30
        GP  = GAT/400.+0.00001
        COL = GP-AINT(GP)
        IF(COL .LT. 0.01) THEN
          IF(GAT .EQ. XCEN) GO TO 12
        ELSE
          IF(GAT .EQ. XCEN) GO TO 12
          YR = YR-1.
        END IF
        GAT = GAT+100.
   10 CONTINUE
   12 CONTINUE
C
      T     = (GAT-1900.)*0.01
C
      OEX(1) = 270.437422 + T*( 307.892    + T*( 0.002525
     &                      + T*( 0.00000189))) + YR*S
      OEX(2) = 334.328019 + T*( 109.032206 + T*(-0.01034444
     &                      + T*(-0.0000125)))  + YR*P
      OEX(3) = 279.696678 + T*(   0.768925 + T*( 0.0003205))
     &                                     + YR*XH
      OEX(4) = 281.220833 + T*(   1.719175 + T*( 0.0004528
     &                      + T*( 0.00000333))) + YR*P1
      OEX(5) = 259.182533 + T*(-134.142397 + T*( 0.00210556
     &                      + T*( 0.00000222))) + YR*XN
C
      DO 100 I=1,5
      ZAT = OEX(I)/360.
      IF(ZAT .LT. 0.) THEN
        OEX(I) = ((ZAT-AINT(ZAT))+1.)*360.
      ELSE
        OEX(I) = (ZAT-AINT(ZAT))*360.
      END IF
      OEX(I) = FLOAT(IFIX(OEX(I)*100.+0.5))*0.01
  100 CONTINUE
C
      XSX  = OEX(1)
      XPX  = OEX(2)
      XHX  = OEX(3)
      XP1X = OEX(4)
      XNX  = OEX(5)
C
```

```
      RETURN
      END
C
C
C
C
      SUBROUTINE VANDUF(SPEED,E,F,ITYPE)
C
C  Computes phases and node factors for tidal constituents.
C
C  References:
C
C  S.P. 98 = US Dept of Commerce, Coast and Geodetic Survey,
C            "Manual of Harmonic Analysis and Prediction
C            of Tides," 1988 Ed.
C  A.M.    = British Admiralty, "Admiralty Manual of Tides,"
C            1941 Ed.
C
C  Common block variables have the following meanings (as
C  best I can tell):
C
C  /LOCAT/ TM and GONL are 0.0 in main program, not defined,
C          but may be related to meridian differences of
C          local time and Greenwich time per Eq. 318, para.
C          223 in S.P. 98
C
C  /FAD/   IPICK is defined in this routine as constituent
C          index based on matching SPEED with array of SPD
C          in NAMES common block
C
C  /VEE/   TML = not defined (may relate to TM in /LOCAT/)
C          CON = defined in this routine
C            U = nu for middle day in radians
C            Q = P         "
C           UI = I         "
C
C  /BOXA/    S = not defined
C           XL = not defined
C           PM = s for beginning day in degrees
C           PL = p          "
C           SL = h          "
C           PS = p1         "
C          PLM = p for middle day in degrees
```

```
C        SKYN = N             "
C          VI = I             "
C           V = nu            "
C          XI = zi            "
C         VPP = nu"           "
C
C  /BOXB/   VP = nu' for middle day in degrees
C           P = P             "
C         AUL = 2*zi - 2*nu - R + 180 , middle day, deg
C         AUM = zi - nu + Q + 90 , middle day, deg
C         CRA = 1/Ra  for L(2) constituent
C         CQA = 1/Qa  for M(1) constituent
C
C  /BOXS/   AW = w   obliquity of ecliptic (max. declination of Sun)
C          AI = i   inclination of Moon's orbit to ecliptic plane
C          AE = e   eccentricity of Moon's orbit
C         AE1 = e1  eccentricity of Earth's orbit
C         ASP = S'  Solar factor
C
C  Order of constituents is same as in  NAMES  common.
C
C  NOTE:  Constituents marked with * have phases shifted
C  by 180 deg from what is given in S.P. 98.  These are coded
C  with the option of adding 180 deg by setting ISHIFT = 1.
C  Original code is retained by setting ISHIFT = 0.
C
      DOUBLE PRECISION     SPEED,  SPD(37)
      DIMENSION          MS(37)
C
      COMMON  /LOCAT/    TM, GONL
      COMMON    /FAD/ IPICK
      COMMON    /VEE/   TML, CON,   U,   Q,  UI
      COMMON   /BOXA/     S,  XL,  PM,  PL,  SL,  PS,  PLM,
     &               SKYN,  VI,   V,  XI, VPP
      COMMON   /BOXB/    VP,   P, AUL, AUM, CRA, CQA
      COMMON   /BOXS/    AW,  AI,  AE, AE1, ASP
      COMMON /SPEEDS/   SPD
      COMMON   /MMSS/    MS
C
      ISHIFT=0             ! if 1, flips *'d constituents by 180 deg
C
      CON  = SL+TML         ! h, beginning day + TML
      C5AW = COS(0.5*AW)    ! cos(w/2)
```

```
      C5AI = COS(0.5*AI)    ! cos(i/2)
      C5UI = COS(0.5*UI)    ! cos(I/2)
      SAW  = SIN(    AW)    ! sin w
      SAI  = SIN(    AI)    ! sin i
      SUI  = SIN(    UI)    ! sin I
C
      DO 600 J=1,37
        IPICK = J
        IF(SPEED .EQ. SPD(J)) GO TO 610
  600 CONTINUE
      WRITE(*,'(2X,"SPEED MATCH NOT FOUND IN VANDUF -> QUIT")')
      WRITE(*,'(2X,"SPEED = ",F12.7)') SPEED
      STOP
  610 CONTINUE
C
      GO TO (  1,  2,  3,  4,  5,  6,  7,  8,  9, 10,
     &        11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
     &        21, 22, 23, 24, 25, 26, 27, 28, 29, 30,
     &        31, 32, 33, 34, 35, 36, 37), IPICK
C*****************************************************************
C  CONSTITUENT CALCULATION CODES
C*****************************************************************
C
C  M(2): term A39, p. 22 and Eq. 70, 78, S.P. 98
C
    1 E = 2.*(CON-PM+XI-V)
      F = (C5AW**4)*(C5AI**4)/(C5UI**4)
      GO TO 800
C
C  N(2): term A40, p. 22 and Eq. 70, 78, S.P. 98
C
    2 E = 2.*(CON+XI-V)-3.*PM+PL
      F = (C5AW**4)*(C5AI**4)/(C5UI**4)
      GO TO 800
C
C  S(2): p. 39, S.P. 98
C
    3 E = 2.*TML
      F = 1.
      GO TO 800
C
C  *O(1): term A14, p. 21 and Eq. 67, 75, S.P. 98
C
```

```
    4 E = CON-V-2.*(PM-XI)-90.+ISHIFT*180.
      F = SAW*(C5AW**2)*(C5AI**4)/(SUI*(C5UI**2))
      GO TO 800
C
C  *K(1): Eq. 222 and 227, p. 45, S.P. 98
C
    5 E = CON-VP+90.+ISHIFT*180.
      F = SQRT(0.8965*(SIN(2.*UI)**2)+0.6001*SIN(2.*UI)*COS(U)
     &                                    +0.1006)
      F = 1./F
      GO TO 800
C
C  K(2): Eq. 230 and 235, p. 46, S.P. 98
C
    6 E = 2.*CON-VPP
      F = SQRT(19.0444*(SUI**4)+2.7702*(SUI**2)*COS(2.*U)+0.0981)
      F = 1./F
      GO TO 800
C
C  L(2): Term A41, p. 22, Eq. 212, p. 44, and Eq. 70, 78, and 215,
C       S.P. 98
C
    7 E = 2.*CON-PM-PL+AUL
      F = (C5AW**4)*(C5AI**4)/(CRA*(C5UI**4))
      GO TO 800
C
C  2N(2): Term A42, p. 22, and Eq. 70, 78, S.P. 98
C
    8 E = 2.*(CON+XI-V+PL)-4.*PM
      F = (C5AW**4)*(C5AI**4)/(C5UI**4)
      GO TO 800
C
C  R(2): Term B47, p. 39, and para. 118, p. 40, S.P. 98
C
    9 E = SL-PS+180.+2.*TML
      F = 1.
      GO TO 800
C
C  T(2): Term B40, p. 39, and para. 118, p. 40, S.P. 98
C
   10 E = 2.*TML-(SL-PS)
      F = 1.
      GO TO 800
```

```
C
C  Lambda(2): Term A44, p. 22, and Eq. 70, 78, S.P. 98
C
  11 E = 2.*(CON+XI-V-SL)-PM+PL+180.
     F = (C5AW**4)*(C5AI**4)/(C5UI**4)
     GO TO 800
C
C  Mu(2): Term A45, p. 22, and Eq. 70, 78, S.P. 98
C
  12 E = 2.*(CON+XI-V+SL)-4.*PM
     F = (C5AW**4)*(C5AI**4)/(C5UI**4)
     GO TO 800
C
C  Nu(2): Term A43, p. 22, and Eq. 70, 78, S.P. 98
C
  13 E = 2.*(CON+XI-V+SL)-3.*PM-PL
     F = (C5AW**4)*(C5AI**4)/(C5UI**4)
     GO TO 800
C
C  *J(1): Term A24, p. 22, and Eq. 68, 76, S.P. 98
C
  14 E = CON+PM-PL-V+90.+ISHIFT*180.
     F = SIN(2.*AW)*(1.-1.5*(SAI**2))/SIN(2.*UI)
     GO TO 800
C
C  *M(1): Eq. 201, p. 42, and Eq. 206, p. 43, S.P. 98
C
  15 E = CON-PM+AUM+ISHIFT*180.
     F = SAW*(C5AW**2)*(C5AI**4)/(CQA*SUI*(C5UI**2))
     GO TO 800
C
C  *OO(1): Term A31, p. 22, and Eq. 69, 77, S.P. 98
C
  16 E = CON-V+2.*(PM-XI)+90.+ISHIFT*180.
     F = SAW*(SIN(0.5*AW)**2)*(C5AI**4)/(SUI*(SIN(0.5*UI)**2))
     GO TO 800
C
C  P(1): Term B14, p. 39, and para. 118, p. 40, S.P. 98
C
  17 E = TML+270.-SL
     F = 1.
     GO TO 800
C
```

```
C  *Q(1): Term A15, p. 21, and Eq. 67, 75, S.P. 98
C
  18 E = CON-V-3.*PM+2.*XI+PL-90.+ISHIFT*180.
     F = SAW*(C5AW**2)*(C5AI**4)/(SUI*(C5UI**2))
     GO TO 800
C
C  *2Q(1): Term A17, p. 21, and Eq. 67, 75, S.P. 98
C
  19 E = CON-V-4.*PM+2.*XI+2.*PL-90.+ISHIFT*180.
     F = SAW*(C5AW**2)*(C5AI**4)/(SUI*(C5UI**2))
     GO TO 800
C
C  *Rho(1): Term A18, p. 21, and Eq. 67, 75, S.P. 98
C
  20 E = CON-V-3.*PM+2.*XI-PL+2.*SL-90.+ISHIFT*180.
     F = SAW*(C5AW**2)*(C5AI**4)/(SUI*(C5UI**2))
     GO TO 800
C
C  M(4): para. 139, p. 47, S.P. 98
C
  21 E = 4.*(CON-PM+XI-V)
     F = ((C5AW**4)*(C5AI**4)/(C5UI**4))**2
     GO TO 800
C
C  M(6): para. 139, p. 47, S.P. 98
C
  22 E = 6.*(CON-PM+XI-V)
     F = ((C5AW**4)*(C5AI**4)/(C5UI**4))**3
     GO TO 800
C
C  M(8): para. 139, p. 47, S.P. 98
C

  23 E = 8.*(CON-PM+XI-V)
     F = ((C5AW**4)*(C5AI**4)/(C5UI**4))**4
     GO TO 800
C
C  S(4): para. 139, p. 47, S.P. 98
C
  24 E = 4.*TML
     F = 1.
     GO TO 800
C
```

```
C  S(6): para. 139, p. 47, S.P. 98
C
  25 E = 6.*TML
     F = 1.
     GO TO 800
C
C  *M(3): Term A82, p. 35, and para. 106, p. 35, S.P. 98
C
  26 E = 3.*(CON-PM+XI-V)+180.+ISHIFT*180.
     F = (C5AW**6)*(C5AI**6)/(C5UI**6)
     GO TO 800
C
C  *S(1): para. 119, p. 40, S.P. 98
C
  27 E = TML+180.+ISHIFT*180.
     F = 1.
     GO TO 800
C
C  *MK(3): p. 68, A.M., M(2)+K(1) interaction
C
  28 E = 2.*(CON-PM+XI-V)+(CON-VP+90.)+ISHIFT*180.
     F = ((C5AW**4)*(C5AI**4)/(C5UI**4))
     F = F/SQRT(0.8965*(SIN(2.*UI)**2)+0.6001*SIN(2.*UI)*COS(U)
    &                                 +0.1006)
     GO TO 800
C
C  *2MK(3): p. 68, A.M., assume M(4)-K(1) interaction, sign is unsure
C
  29 E = 4.*(CON-PM+XI-V)-(CON-VP+90.)+ISHIFT*180.
     F = ((C5AW**4)*(C5AI**4)/(C5UI**4))**2
     F = F/SQRT(0.8965*(SIN(2.*UI)**2)+0.6001*SIN(2.*UI)*COS(U)
    &                                 +0.1006)
     GO TO 800
C
C  MN(4): p. 68, A.M., M(2)+N(2) interaction
C
  30 E = 4.*(CON+XI-V)+PL-5.*PM
     F = ((C5AW**4)*(C5AI**4)/(C5UI**4))**2
     GO TO 800
C
C  MS(4): p. 67, A.M., M(2)+S(2) interaction
C
  31 E = 2.*(CON-PM+XI-V)+2.*TML
```

```
      F = (C5AW**4)*(C5AI**4)/(C5UI**4)
      GO TO 800
C
C  2SM(2): p. 68, A.M., 2M(2)-S(2) interaction
C
   32 E = 4.*TML-2.*(CON-PM+XI-V)
      F = (C5AW**4)*(C5AI**4)/(C5UI**4)
      GO TO 800
C
C  Mf: Term A6, p. 21, and Eq. 66, 74, S.P. 98
C
   33 E = 2.*(PM-XI)
      F = (SAW**2)*(C5AI**4)/(SUI**2)
      GO TO 800
C
C  MSf: Term A5, p. 21, and Eq. 65, 73, S.P. 98; p. 67, A.M.,
C       S(2)-M(2) interaction
C       Note:  F was redefined to be consistent with S.P. 98,
C              old F is commented out...
C
   34 E = 2.*TML-2.*(CON-PM+XI-V)
C     F = (C5AW**4)*(C5AI**4)/(C5UI**4)
      F = ((2./3.-(SAW**2))*(1.-1.5*(SAI**2)))/(2./3.-(SUI**2))
      GO TO 800
C
C  Mm: Term A2, p. 21, and Eq. 65, 73, S.P. 98
C
   35 E = PM-PL
      F = ((2./3.-(SAW**2))*(1.-1.5*(SAI**2)))/(2./3.-(SUI**2))
      GO TO 800
C
C  Sa: para. 119, p. 40, S.P. 98
C
   36 E = SL
      F = 1.
      GO TO 800
C
C  Ssa: Term B6, p. 39, S.P. 98
C
   37 E = 2.*SL
      F = 1.
      GO TO 800
C****************************************************************
```

```
C  END OF CONSTITUENT CALCULATION CODES
C***************************************************************
C
  800 CONTINUE
C
      IF(ITYPE .EQ. 2) THEN
        E = E + FLOAT(MS(IPICK))*GONL-SPD(IPICK)*TM/15.
      ELSE IF(ITYPE .EQ. 1) THEN
        ZAT = E/360.                   !# OF CIRCLES
        IF(ZAT.LT.0.) THEN
          E = (ZAT-AINT(ZAT)+1.)*360.      !360 + MOD(E,360)
        ELSE
          E = (ZAT-AINT(ZAT))*360.        !MOD(E,360)
        END IF
      END IF
C
      F = 1./F
C
      RETURN
      END
C
C
C
C
      SUBROUTINE TABLE6(VI,V,XI,VP,VPP,CIG,CVX,CEX,PVC,
     &                          PVCP,ANG,AN,AT)
C
C  Using as input the longitude of the moon's node N (here
C  called ANG) and the astronomical constants w, i, e, e1
C  and S' (here called AW, AI, AE, AE1 and ASP, respectively,
C  and input through the common block BOXS) from the beginning
C  of subroutine ASTRO, computes the 5 astronomical entities
C  listed in Table 6 of S.P. 98 and used for tidal computations,
C  specifically:
C
C    VI = inclination of moon's orbit to celestial equator (I)
C         in degrees to nearest 0.01 deg
C     V = right ascension or longitude in celestial equator of
C         moon's orbit (nu) in degrees to nearest 0.01 deg
C    XI = longitude in moon's orbit of the lunar intersection
C         (zi) to nearest 0.01 deg
C    VP = nu' of Eq. 224 in S.P. 98 in degrees
C   VPP = 2*nu" of Eq. 232 in S.P. 98 in degrees
```

```
C
C  Also returned are:
C
C   CIG = VI (I) in radians
C   CVX = V (nu) in radians
C   CEX = XI (zi) in radians
C   PVC = VP (nu') in radians
C  PVCP = VPP (2*nu") in radians
C    AN = ANG (N) in radians
C    AT = CIG identically, i.e., redundancy
C
      COMMON /BOXS/ AW, AI, AE, AE1, ASP
C
      PI   = 4.*ATAN(1.)               !YOU KNOW, PI
      DTR  = 180./PI                  !DEG./RAD., 57.295780
      RTD  = PI/180.                  !RAD./DEG., 0.0174533
C
C  Initialize returned values
C
      V    = 0.0       !nu, right ascension
      XI   = 0.0       !zi, long. in moon's orbit of intersection
      VP   = 0.0       !nu' of Eq. 224
      VPP  = 0.0       !2*nu" of Eq. 232
      AN   = ANG*RTD   !N in radians
C
      AX   = ANG                                !redundant replacement
      EYE  = COS(AI)*COS(AW)-SIN(AI)*SIN(AW)*COS(AN)  !cos(I)
      C9   = DTR*ACOS(EYE)                          !I in deg
      VI   = FLOAT(IFIX(C9*100.+0.5))*0.01          !I to nearest 0.01 deg
      CIG  = RTD*VI                                 !VI in radians
      AT   = CIG                                    !redundant CIG
C
C  Special condition checks
C
      IF( CIG .EQ.   0. ) GO TO 230
      IF(  AX .EQ.   0. ) GO TO 230
      IF(  AX .EQ. 180. ) GO TO 230
C
C  Spherical trigonometry to get nu (Fig. 1, Pg. 6, S.P. 98)
C
      VXXE = SIN(AI)*SIN(AN)
      VXXN = COS(AI)*SIN(AW)+SIN(AI)*COS(AW)*COS(AN)
      IF( VXXE .EQ. 0. ) GO TO 201
```

```
      IF( VXXN .EQ. 0. ) GO TO 201
      VXX  = VXXE/VXXN                    !tan(nu)
      C10  = DTR*ATAN(VXX)                !nu in deg.
      V    = FLOAT(IFIX(C10*100.+0.5))*0.01       !nu to 0.01 deg
      IF( V .GT. 0. .AND. AX .GT. 180. ) V = -V    !- for 180 < N < 360
C
  201 CONTINUE
C
      CVX  = RTD*V                        !nu in radians
C
C  Spherical trigonometry to get zi (Fig. 1, Pg. 6, S.P. 98)
C
      TERM = SIN(AI)*COS(AW)/SIN(AW)
      EXX  = TERM*SIN(AN)/COS(AN) + (COS(AI)-1.)*SIN(AN)
      EZZ  = TERM+COS(AI)*COS(AN) + (SIN(AN)**2)/COS(AN)
      IF( EXX .EQ. 0. )    GO TO 202
      IF( EZZ .EQ. 0. )    GO TO 202
      EXEZ = EXX/EZZ                      !tan(zi)
      IF( EXEZ .GT. 3450. ) GO TO 202
      C11  = DTR*ATAN(EXEZ)               !zi in deg.
      XI   = FLOAT(IFIX(C11*100.+0.5))*0.01       !zi to 0.01 deg
      IF( XI .GT. 0. .AND. AX .GT. 180.) XI = -XI   !- if 180 < N < 360
C
  202 CONTINUE
C
      CEX  = RTD*XI                       !zi in radians
C
C  From paragraph 133 of S.P. 98 to find nu':
C
      A22  = (0.5+0.75*(AE**2))*SIN(2.*CIG)       !lunar coeff. A of Eq. 216
      B22  = (0.5+0.75*(AE1**2))*ASP*SIN(2.*AW)     !solar coeff. B of Eq. 217
      VPXE = A22*SIN(CVX)                 !numerator in Eq. 224
      VPXN = A22*COS(CVX)+B22             !denominator in Eq. 224
      IF( VPXE .EQ. 0. )   GO TO 203
      IF( VPXN .EQ. 0. )   GO TO 203
      VPX  = VPXE/VPXN                    !tan(nu')
      IF( VPX .GT. 3450. ) GO TO 203
      VP   = DTR*ATAN(VPX)                !nu' in degrees
      IF( VP .GT. 0. .AND. AX .GT. 180.) VP = -VP   !- if 180 < N < 360
C
  203 CONTINUE
C
      PVC  = RTD*VP                       !nu' in radians
```

```
C
C  From paragraph 135 in S.P. 98 for 2*nu":
C
      A47  = (0.5+0.75*(AE**2))*(SIN(CIG)**2)       !lunar coeff. A of Eq. 218
      B47  = (0.5+0.75*(AE1**2))*ASP*(SIN(AW)**2)   !solar coeff. B of Eq. 219
      VPYE = A47*SIN(2.*CVX)                        !numerator in Eq. 232
      VPYN = A47*COS(2.*CVX)+B47                    !denominator in Eq. 232
      IF( VPYE .EQ. 0. )   GO TO 204
      IF( VPYN .EQ. 0. )   GO TO 204
      VPY  = VPYE/VPYN                              !tan(2*nu")
      IF( VPY .GT. 3450. ) GO TO 204
      VPP  = DTR*ATAN(VPY)                          !2*nu" in degrees
      IF( VPP .GT. 0. .AND. AX .GT. 180.) VPP=-VPP  !- if 180 < N < 360
C
  204 CONTINUE
C
      PVCP = RTD*VPP                                !2*nu" in radians
C
C  Special condition jump address
C
  230 CONTINUE
C
      RETURN
      END
C
C
C
C
      SUBROUTINE NAME(SPDD,ITAG,ISUB,INUM,ICODE)
C*********************************************************
C  ROUTINE IDENTIFIES A CONSTITUENT BY ITS SPEED          *
C  (ICODE=1), ITS NAME LABEL (ICODE=2), OR ITS CONSTITU-  *
C  ENT NUMBER (ICODE=3), AND MAKES IT AVAILABLE FOR       *
C  LABELING.  IT ALSO DETERMINES THE SUBSCRIPT OF THE     *
C  CONSTITUENT.  THE ORDER OF THE CONSTITUENT SPEEDS IS:  *
C                                                         *
C    M(2)    N(2)    S(2)    O(1)      K(1)     K(2)      *
C    L(2)   2N(2)    R(2)    T(2)  LAMBDA(2)    MU(2)     *
C   NU(2)    J(1)    M(1)   OO(1)      P(1)     Q(1)      *
C   2Q(1)  RHO(1)    M(4)    M(6)      M(8)     S(4)      *
C    S(6)    M(3)    S(1)   MK(3)    2MK(3)    MN(4)      *
C   MS(4)  2SM(2)   MF      MSF        MM       SA        *
C  SSA                                                    *
```

```
C*****************************************************************
      CHARACTER*10     LABLE(37), ITAG
      DIMENSION           IP(37)
      DOUBLE PRECISION   SPD(37), SPDD
C
      COMMON   /MMSS/    IP
      COMMON /SPEEDS/   SPD
      COMMON  /NAMES/ LABLE
C
    1 FORMAT(10X,'Speed constituent ',F12.7,' not in list')
    2 FORMAT(10X,'Constituent named ',A10,' not in list')
    3 FORMAT(10X,'Constituent number ',I4,' not in list')
C
      IF(ICODE.EQ.1) THEN              !SEARCH BY SPEED
        DO 100 J=1,37
          IF(SPDD.NE.SPD(J)) THEN
            GO TO 100
          ELSE
            ITAG=LABLE(J)
            ISUB=IP(J)
            INUM=J
            GO TO 400
          END IF
  100   CONTINUE
        WRITE(*,1) SPDD
C
      ELSE IF(ICODE.EQ.2) THEN         !SEARCH BY NAME
        DO 200 I=1,37
          IF(ITAG.NE.LABLE(I)) THEN
            GO TO 200
          ELSE
            SPDD=SPD(I)
            ISUB=IP(I)
            INUM=I
            GO TO 400
          END IF
  200   CONTINUE
        WRITE(*,2) ITAG
C
      ELSE IF(ICODE.EQ.3) THEN         !SEARCH BY NUMBER
        K=INUM
        IF(K.GT.0 .AND. K.LE.37) THEN
          ITAG=LABLE(K)
```

```
          SPDD=SPD(K)
          ISUB=IP(K)
          GO TO 400
        END IF
        WRITE(*,3) INUM
      END IF
C
      WRITE(*,'("****EXECUTION STOPPED IN SUBROUTINE NAME****")')
      STOP
C
  400 CONTINUE
C
      RETURN
      END
C
C
C
C
      BLOCK DATA CONSTS
C*********************************************************
C  37 CONSTITUENT ELEMENTS IS NOAA STANDARD AS OF PEARL   *
C  HARBOR DAY 1987                                        *
C*********************************************************
      CHARACTER*10     LABLE(37)
      INTEGER          MS(37)
      DOUBLE PRECISION   SPD(37)
C
      COMMON /SPEEDS/   SPD
      COMMON  /NAMES/ LABLE
      COMMON   /MMSS/    MS
C
C  Tidal constituent speeds in degrees per hour
C
      DATA        SPD/
     &   28.9841042D0,  28.4397295D0,  30.0000000D0,  13.9430356D0,
     &   15.0410686D0,  30.0821373D0,  29.5284789D0,  27.8953548D0,
     &   30.0410667D0,  29.9589333D0,  29.4556253D0,  27.9682084D0,
     &   28.5125831D0,  15.5854433D0,  14.4966939D0,  16.1391017D0,
     &   14.9589314D0,  13.3986609D0,  12.8542862D0,  13.4715145D0,
     &   57.9682084D0,  86.9523127D0, 115.9364169D0,  60.0000000D0,
     &   90.0000000D0,  43.4761563D0,  15.0000000D0,  44.0251729D0,
     &   42.9271398D0,  57.4238337D0,  58.9841042D0,  31.0158958D0,
     &    1.0980331D0,   1.0158958D0,   0.5443747D0,   0.0410686D0,
```

```
     &     0.0821373D0/
C
C  Constituent subscripts
C
      DATA         MS/
     &             2,          2,          2,          1,
     &             1,          2,          2,          2,
     &             2,          2,          2,          2,
     &             2,          1,          1,          1,
     &             1,          1,          1,          1,
     &             4,          6,          8,          4,
     &             6,          3,          1,          3,
     &             3,          4,          4,          2,
     &             0,          0,          0,          0,
     &             0/
C
C  Constituent names
C
      DATA      LABLE/
     &   'M(2)      ' , 'N(2)      ' , 'S(2)      ' , 'O(1)      ' ,
     &   'K(1)      ' , 'K(2)      ' , 'L(2)      ' , '2N(2)     ' ,
     &   'R(2)      ' , 'T(2)      ' , 'Lambda(2) ' , 'Mu(2)     ' ,
     &   'Nu(2)     ' , 'J(1)      ' , 'M(1)      ' , 'OO(1)     ' ,
     &   'P(1)      ' , 'Q(1)      ' , '2Q(1)     ' , 'Rho(1)    ' ,
     &   'M(4)      ' , 'M(6)      ' , 'M(8)      ' , 'S(4)      ' ,
     &   'S(6)      ' , 'M(3)      ' , 'S(1)      ' , 'MK(3)     ' ,
     &   '2MK(3)    ' , 'MN(4)     ' , 'MS(4)     ' , '2SM(2)    ' ,
     &   'Mf        ' , 'Msf       ' , 'Mm        ' , 'Sa        ' ,
     &   'Ssa       ' /
C
       END
c***********************************************************************
      INTEGER FUNCTION LSTGDCHR(NAMESUB)
c
      CHARACTER*(*)     NAMESUB
      INTEGER     II,CLEN,INDEX
c
      CLEN = LEN(NAMESUB)
      DO 100 II = CLEN,1,-1
100   IF( ICHAR(NAMESUB(II:II)) .GT. 60 .AND.
     1      ICHAR(NAMESUB(II:II)) .LT. 127 )GOTO 101
101   INDEX=II
c
```

```
      LSTGDCHR = INDEX
      RETURN
c
      END
```

APPENDIX C

CODES FOR THE OCEANRDS REFRACTION/DIFFRACTION MODEL

Appendix C:
Codes for the OCEANRDS Refraction/Diffraction Model

There is a family of OCEANRDS codes, with each specific to particular grid domains. For the back refraction problems and other far field applications the **oceanrds_socal.for** version is used on a 2,405 x 4,644 raster formatted grid that encompasses the entire Southern California Bight. The input parameters output files which are required by **oceanrds_socal.for** are

graham_m.grd *.........* (name.grd) bathymetry input file
-1.0.........................* (gis) if data is parsed GIS data gis= -1.0, if NOS data gis=1.0
1.........................* wave exposure 1=west, 2=north, 3=east, 4=south (icoast)
0.0.........................* sea level adjustment MSL meters (sealev) (+ = deeper water)
77.5.........................* outer grid dimensions in meters perpendicular to coast (sx)
92.6.........................* outer grid dimensions in meters parallel to coast (sy)
4644.........................* number of grid cells in from deep water perpendicular to coast raster (nx)
2405.........................* number of grid cells along coast from top edge (ny)
17.0.........................* wave period in seconds (persw)
270.0.........................* wave direction degress clockwise from true north (asw)
10.0.........................* wave height meters (hsw)

```
c*******************************************************************
c
c       Ocean refraction - diffraction module oceanrds_socal.for
c
c    programmed to run on full socal elevation data (graham_m.grd)
c    extracted from GIS grid  by read_graham_bathy_full.for
c                          10 oct, 2002
c
c*******************************************************************
c   This program is the first program in a 2 program series to treat
c   the distribution of wave heights, angles, wave numbers from the
c   combined effect of a wave field containing two distinct periods
c   and/or directions. The second program (windwave.f) must also be run
c   in order for the ultimate output file to be properly name for use
c   by the other modules even if there is 0 energy in the second band.
c   It solves a parabolic approximation to the mild slope equation
c   for the transmitted field of a linear wave.  Outputs wave height,
c   wave number, and wave angle for each grid point in a 2405 x 4644
c   grid array with 3 second x 3 second spacing.  Bathymetry is read from
c   a formatted 2405 x 4644 real number array called 'ifile'.grd,
c   created by the oceanbat.f module, where'ifile' corresponds to
c   the inputted site name.  Program uses "oceanrd.inp" input file.
c   All output files are named 'ifile' with different extensions,
c   ie. 'ifile'.wh1.
c*******************************************************************
c*******************************************************************
c
            parameter (max=10000000)
        character  name*8,ifile*12
        character ofile1*12,ofile2*12,ofile3*12,ofile4*12
        character ofile5*12,ofile6*12,ofile7*12
       dimension  ccg(max),di(max),hab(max),rlb(max),dib(max),ih(max)
       dimension depth(4644,4644),wht(4644,4644)
       dimension ang(4644,4644)
        dimension depthold(4644,4644)
        real kbar(2,max),kave
c
        complex aa(max),bb(max),cc(max),dd(max),uu(max),aprev(max)
        complex c3,t1,t2,t3,f,alast(max),aphys(max),mim,mip
c
        common /grid/ ny,sy,sx,dely,delx,ndely,freq,dcon,ify,ifx,tide
              common /cut/ dc
c
```

```
      pi=acos(-1.)
c
c   open parameter input file, return error message if missing
c
c
c
      open(2,file='oceanrds_socal.inp',status='old',err=900)
      read(2,'(a)') name
         read(2,*) gis
      read(2,*) icoast
         read(2,*) tide_elev
         read(2,*) sx
      read(2,*) sy
         read(2,*) nx
      read(2,*) ny
         read(2,*) persw
         read(2,*) asw
         read(2,*) hsw
c
      tide=tide_elev
      amp=hsw
      dir=asw
      per=persw
c
         freq=1/per
c
c... change to proper rotation frame, flag if theta not between +/- 45
c
      IF (icoast .EQ. 1) coast=270.0
      IF (icoast .EQ. 2) coast=90.0
      IF (icoast .EQ. 3) coast=0.0
      IF (icoast .EQ. 4) coast=180.0
c
         theta=coast-dir
c
c... set some variables to constant values
c
c       dcon = 1. for nuwave version
       dcon=1.
c
c       cutoff depth (dc) -5.0 meters
       dc=-5.0
c
```

```
c        breaking wave switch (ibreak) (1=on, 0=off)
         ibreak=1
c
c        breaking criteria (bc) 0.5  for wave height = (bc)*depth
         bc=0.5
c
c        lateral b.c.: ibc=(0) transmitting, ibc=(1) reflective
         ibc=0
c        if trans: (isn=0) straight snell, isn=(1) kirby's improved
         isn=1
c        idf=(0) small angle diff, idf=(1) large angle dif
         idf=1
c
c
           write(*,'(5(/),10x,a)')
     &'  KIRBY HIGHER ORDER REFRACTION-DIFFRACTION PROGRAM '
           write(*,'(/,10x,a)')
     &'- based on the parabolic equation method (PEM) of solving'
           write(*,'(10x,a,5(/))')
     &'  the mild-slope equation. '
c
           nn=8
           do 3 m=8,1,-1
3          if(name(m:m).eq.' ') nn=m-1
c
c
c... open grid file
c
           write(ifile,'(a,a)') name(:nn),'.grd'
        open(20,file=ifile,status='old',err=900)
c
c
c... open breaker ix,iy,wave height, wave angle, depth file
        write(ofile7,'(a,a)') name(:nn),'.bra'
         open(39,file=ofile7)
c
c
c... open sea level corrected ascii bathymetry file
        write(ofile1,'(a,a)') name(:nn),'.dep'
        open(31,file=ofile1)
c
c
c... open ascii wave number file
```

```
      write(ofile2,'(a,a)') name(:nn),'.wvn'
      open(32,file=ofile2)
c
c
c
c... open ascii wave height file
      write(ofile3,'(a,a)') name(:nn),'.wh1'
      open(33,file=ofile3)
c
c
c
c... open ascii wave angle file
      write(ofile4,'(a,a)') name(:nn),'.an1'
      open(34,file=ofile4)
c
c
c
c....  create depth array
c
      do 111 j=ny,1,-1
      read(20,*) (depthold(j,i),i=1,nx)
111   continue
      rewind(20)
c
c 10oct02 *******correct for sea level and change sign if GIS data
      do 711 i=1,nx
      do 811 j=1,ny
      depthold(j,i)=(gis*depthold(j,i))+sealev
811   continue
711   continue
c
c......write rotated ascii depth file for internal use in ref/dif
165   format(2405f12.2)
      DO 101 i=1,nx
      WRITE(31,165)(depthold(j,i),j=1,ny)
101   CONTINUE
      rewind(31)
c
c
      do 119 i=1,nx
      read(31,*) (depth(i,j),j=1,ny)
119   continue
      rewind(31)
```

```
c
c
c... read first depth to initialize offshore boundary
c
            read(31,*) (di(m),m=1,ny)
            ridep=di(ny/2)
            rewind(31)

c.... would like a grid step size on the order of 1/5 wavelength

            call getkcg(10.,wk,cg0)
            wl=2*pi/wk
            ifx=1+sx/(.2*wl)
            ify=1+sy/(.2*wl)
c
            nmx=ifx
            nmy=ify
c
        amp=amp/2.
c
c... calculate dimensions of interpolated grid
c
            dely=sy/ify
            delx=sx/ifx
        ndely=(ny-1)*ify
        ndelx=(nx-1)*ifx
c
c
        call getkcg(ridep,wk0,cg0)
c  wave length (m)
        wl=2*pi/wk0
c  wave frequency (rad/sec)
        sig=2*pi/per
c  radder's correction factor
        do 33 j=1,ndely+1
 33     ccg(j)=sqrt(sig*cg0/wk0)
c
c
             ltype=2
c... open binary wave height file
        write(ofile5,'(a,a)') name(:nn),'.bw1'
         open(9,file=ofile5,form='unformatted')
c
```

```
c... open binary wave angle file
        write(ofile6,'(a,a)') name(:nn),'.ba1'
         open(11,file=ofile6,form='unformatted')
c
c.. initialize depth array
            do 555 nn=1,ndely+1
            hab(nn)=0.
            ih(nn)=0
555         continue
c
c   enter initial condition at x=0
c
        call inbc(amp,wk0,theta,ndely,dely,alast)
c
        do 202 j=1,ndely+1
        kbar(1,j)=wk
202       kbar(2,j)=wk
c
c scale alast as in radder(1978)
        do 32 j=1,ndely+1
        alast(j)=alast(j)*ccg(j)
32        continue
c
c        call wwave(alast,ndely,nmy,dely,kbar)
c
c    solution of the parabolic eqn. by the crank-nicholson formulation
c
c    start x increments
c
        c2=1./2./dely**2
        c3=2.*(0,1)/delx
c
c--------------------------------------------------------------------
c increments in x-direction
c
        ikount=0
        do 100 l=1,ndelx
c
c... write every 10th step to the screen
c
            if(10*((l+1)/10).eq.l+1) then
        print *,' column ',l+1,' of ',ndelx+1
            endif
```

```
c
      lm=l
c
      call intkcg(lm,kbar,ccg,di,ikount)
c
c correction factor
      do 34 j=1,ndely+1
      ccg(j)=sqrt(sig*ccg(j)/kbar(2,j))
34    continue
c-----------------------------------------------------------------------
c increments in y-direction - 1st. round
      do 200 j=2,ndely
c
      kave=(kbar(2,j)+kbar(1,j))/2.
c      kave=kbar(2,j)
c
      if(idf.eq.0) then
c small angle diffraction approximation
c
      t1=c2
      t3=c3*kave
      f=2*kave*(kave-wk)+c3/2*(kbar(2,j)-kbar(1,j))
c
      aa(j-1)=t1
      bb(j-1)=t3-2.*t1+f/2.
      cc(j-1)=t1
      dd(j-1)=-t1*alast(j+1)+(t3+2*t1-f/2)*alast(j)-t1*alast(j-1)
c
      endif
c
      if(idf.eq.1) then
c large angle diffraction approximation
c
      t1=c2/2.*(3.-wk/kave)
      t2=c2*c3/4./kave
      t3=c3*kave
      f=2*kave*(kave-wk)+c3/2*(kbar(2,j)-kbar(1,j))
c
      aa(j-1)=t1+t2
      bb(j-1)=t3-2.*(t1+t2)+f/2.
      cc(j-1)=t1+t2
      dd(j-1)=(t2-t1)*(alast(j+1)+alast(j-1))+(t3-2*(t2-t1)-f/2)
     &    *alast(j)
```

```
      endif
c
      if(j.eq.2) then
c      if(ibc.eq.1) cc(1)=aa(1)+cc(1)
c reflective b.c. : a1=a3
      if(ibc.eq.1) bb(1)=aa(1)+bb(1)
c reflective b.c. : a1=a2
      if(ibc.eq.0) then
c transmitting b.c.
c
      if(isn.eq.0) then
c straight snell
      mip= (0.,1.)*wk*sin(theta)/2.
      mip=(1./dely-mip)/(1./dely+mip)
      else
c  kirby's improved
      mip=(alast(2)-alast(1))/(alast(2)+alast(1))
      mip=(1.-mip)/(1.+mip)
      endif
c
      bb(1)=bb(1)+mip*aa(1)
      endif
      aa(j-1)=(0.,0.)
c
      endif
c
      if(j.eq.ndely) then
c      if(ibc.eq.1) aa(j-1)=aa(j-1)+cc(j-1)
c reflective b.c. : an=an-2
      if(ibc.eq.1) bb(j-1)=bb(j-1)+cc(j-1)
c reflective b.c. : an=an-1
      if(ibc.eq.0) then
c transmitting b.c.
c
      if(isn.eq.0) then
c   straight snell
      mim= (0.,1.)*wk*sin(theta)/2.
      mim=(1./dely+mim)/(1./dely-mim)
      else
c   kirby's improved
      mim=(alast(j+1)-alast(j))/(alast(j+1)+alast(j))
      mim=(1.+mim)/(1.-mim)
      endif
```

```
c
      bb(j-1)=bb(j-1)+mim*cc(j-1)
      endif
      cc(j-1)=(0.,0.)
      endif
c
200   continue
c----------------------------------------------------------------------
c
c              *****   compute the solution  *****
c
      neqs=ndely-1
c
      call tridag(aa,bb,cc,dd,uu,neqs)
c
      do 143 j=2,ndely
143   alast(j)=uu(j-1)
c load alast(j)
c
      if(ibc.eq.1) then
c      alast(1)=alast(3)
c reflective b.c.: a1=a3
c      alast(ndely+1)=alast(ndely-1)
c             : an=an-2
      alast(1)=alast(2)
c reflective b.c.: a1=a2
      alast(ndely+1)=alast(ndely)
c             : an=an-1
      endif
c
      if(ibc.eq.0) then
c transmitting b.c.
c
      alast(1)= mip*alast(2)
      alast(ndely+1)= mim*alast(ndely)
c
      endif
c
c transform back into phys. height
      do 37 j=1,ndely+1
37    aphys(j)=2.*alast(j)/ccg(j)
c
c          *********   check for breaking   *********
```

```
c
            if(ibreak.eq.1) then
        do 54 j=1,ndely+1
        hb=bc*di(j)
        rat=hb/cabs(aphys(j))
              if(di(j).lt.0) rat=0.
        if(rat.lt.1) then
c  save point just before wave first breaks
               if(ih(j).eq.0) then
                          hab(j)=cabs(aprev(j))
                          rlb(j)=(lm-1)/real(ifx)
c  calculate direction before breaking
                            if(j.eq.ndely+1.or.ih(j+1).eq.1) then
            xx=aimag((aprev(j)-aprev(j-1))/(dely*kbar(1,j)*aprev(j)))
                              if(xx.gt.1) xx=1.
                          if(xx.lt.-1.) xx=-1.
                              dib(j)=asin(xx)
                            else
            xx=aimag((aprev(j+1)-aprev(j))/(dely*kbar(1,j)*aprev(j)))
                              if(xx.gt.1) xx=1.
                              if(xx.lt.-1.) xx=-1.
                          dib(j)=asin(xx)
                        endif
                  dib(j)=270.-dib(j)*57.296-rot
                             if(dib(j).lt.0) dib(j)=360+dib(j)
               ih(j)=1
               endif
        aphys(j)=rat*aphys(j)
        alast(j)=rat*alast(j)
            endif
54          continue
        endif
c
c  ****************************************************************
c
c... writing wave field
            if(nmx*(lm/nmx).eq.lm.or.lm.eq.1) then
          call wwave(aphys,ndely,nmy,dely,kbar)
            endif
c
            do 55 nn=1,ndely+1
            aprev(nn)=aphys(nn)
55          continue
```

```
c
100     continue
c
c... output prebreak heights and directions
           open(24,file='break.dat')
           write(24,'(/,18x,a)') '         REFRACTION - DIFFRACTION         '
           write(24,'(/,18x,a)') ' WAVE HEIGHT AND DIRECTION BEFORE BREAKING'
           write(24,'(/,15x,a,a,a,f5.1,a,f4.1)')
     &  ' site: ',name,'   direction: ',dir,'  period: ',per
           write(24,'(/,20x,a,6x,a,6x,a,9x,a,//)')
     &'row',' H (m) ','col','dir'
           m=1
           write(24,'(18x,i4,f12.2,f12.1,f12.1)') m,hab(m),rlb(m),dib(m)
        alphab=dib(m)-shor
        irlb=NINT(rlb(m))+1
        breakd=(5.0/4.0)*hab(m)
***rasterisze break point file m.....ixbra
        ixbra=2405-m
        write(39,'(18x,i4,i4,f12.3,f12.1,f12.3)') ixbra,irlb,hab(m),
     &alphab,breakd
           if(nmy.eq.1) then
           is=2
           else
           is=nmy
           endif
           mm=1
           do 888 m=is,ndely,nmy
           mm=mm+1
           write(24,'(18x,i4,f12.2,f12.1,f12.1)') mm,hab(m),rlb(m),dib(m)
        alphab=dib(m)-shor
        irlb=NINT(rlb(m))+1
        breakd=(5.0/4.0)*hab(m)
        ixbra=2405-mm
        write(39,'(18x,i4,i4,f12.3,f12.1,f12.3)')ixbra,irlb,hab(m),
     &alphab,breakd
888     continue
c
c
        rewind (9)
        rewind (11)
c
c......create wave height and wave angle arrays
        do 114 i=1,nx
```

```
                read(9) (wht(i,j),j=1,ny)
                read(11) (ang(i,j),j=1,ny)
114     continue
c
c.... write array values to:
c.... ascii depth, wave number, wave angle, wave height grid files
c
c
c2002    format(4644f12.2)
c
cccc Comment out old way of writing arrays
cccccccccc        do 134 i=1,nx
cccccccccc        write(33,2002) (wht(i,j),j=1,ny)
cccccccccc        write(34,2002) (ang(i,j),j=1,ny)
cccccccccc134     continue
c
CCCCCC Raster way of writing arrays!!!!!!!!!!!!!
       DO 176 j=ny,1,-1
         write(33,6666) (wht(i,j),i=1,nx)
         write(34,6666) (ang(i,j),i=1,nx)
  176  CONTINUE
6666     format(4644f12.2)
6667     format(4644f12.6)
c
c
            go to 901
900      write(*,'(20x,a,a,a)') '**** error **** ',ifile,' missing'
            write(*,'(20x,a)') '    press any key to continue '
            read(*,'(a)') idum
901     continue
        stop
        end
c
c-------------------------------------------------------------------
c
        subroutine tridag(a,b,c,r,u,n)
c
        parameter (nmax=250002)
        complex gam(nmax),a(n),b(n),c(n),r(n),u(n),bet
c
        bet=b(1)
        u(1)=r(1)/bet
        do 11 j=2,n
```

```
      gam(j)=c(j-1)/bet
      bet=b(j)-a(j)*gam(j)
      if(bet.eq.0)pause
      u(j)=(r(j)-a(j)*u(j-1))/bet
11    continue
      do 12 j=n-1,1,-1
      u(j)=u(j)-gam(j+1)*u(j+1)
12    continue
      return
      end
c
c---------------------------------------------------------------------
c
      subroutine wwave(aphys,ndely,nmy,dely,kbar)
c
c  compute & write wave height and direction, initial breaking point
c    imax= max grid size nmax=max no. of steps
c
      parameter (nmax=45000000,imax=4500000)
         dimension amod(nmax),alfa(nmax)
         real kbar(2,nmax)
         complex aphys(nmax)
c
         nky=0
      do 350 j=1,ndely+1
         if(nmy*(j/nmy).eq.j.or.j.eq.1) then
         nky=nky+1
c
          amod(nky)=cabs(aphys(j))
          if(amod(nky).gt..05) then
          if(j.eq.ndely+1) then
          xx=aimag((aphys(j)-aphys(j-1))/(dely*kbar(2,j)*aphys(j)))
          if(xx.gt.1) xx=1.
          if(xx.lt.-1.) xx=-1.
          alfa(nky)=asin(xx)
          else
          xx=aimag((aphys(j+1)-aphys(j))/(dely*kbar(2,j)*aphys(j)))
          if(xx.gt.1) xx=1.
          if(xx.lt.-1.) xx=-1.
          alfa(nky)=asin(xx)
          endif
          else
          alfa(nky)=0
```

```
            endif
            alfa(nky)=alfa(nky)*57.296
           endif
350     continue
c
         write(9) (amod(jj),jj=1,nky)
         write(11) (alfa(jj),jj=1,nky)
c
        return
        end
c
c---------------------------------------------------------------------
c
        subroutine inbc(amp,wk,theta,ndely,dely,aa)
c
        complex aa(*)
c
        real ky
        pi=acos(-1.)
        theta=theta*pi/180.
        ky=wk*sin(theta)
c
        do 3 j=1,ndely+1
        ar=amp*cos(ky*(j-1)*dely)
        ai=amp*sin(ky*(j-1)*dely)
c       ar=(cos(ky*(j-1)*dely)+cos(-ky*(j-1)*dely))/2.
c 2 identical
c       ai=(sin(ky*(j-1)*dely)+sin(-ky*(j-1)*dely))/2.
c waves summed
        aa(j)=cmplx(ar,ai)
3       continue
c
        return
        end
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
        subroutine getkcg(d,k,cg)
c
c....  depth (meters) and frequency are input
c....  returns wave number (2pi/l) and group velocity
c
        real k
```

```
      common /grid/ ny,sy,sx,dely,delx,ndely,f,dcon,ify,ifx,tide
      data tpi/6.2831853/
c
      sig=tpi*f
      a=d*sig*sig/9.81
      if(a.ge.1) then
            yhat=a*(1+1.26*exp((-1.84)*a))
            t=exp((-2)*yhat)
            aka=a*(1+2*t*(1+t))
      else
            aka=sqrt(a)*(1+a/6.*(1+a/5.))
      endif
      k=aka/d
      x=2*k*d
      cg=tpi*(f/k)*.5*(1+x/sinh(x))
      return
      end
c
c*****************************************************************
c
      subroutine intkcg(lm,kbar,ccg,di,ikount)
c
         parameter(max=45000002)
      real kbar(2,max)
      dimension ccg(max),d(2,max),di(max)
c
      save d
           common /cut/ dc
      common /grid/ ny,sy,sx,dely,delx,ndely,f,dcon,ify,ifx,tide
      data istart,ncol,eps/1,0,.0000001/
c
c... if this is the first call, load first two columns
c
      go to (1,3) istart
1       istart=2
c... read in first 2  columns
           do 777 i=1,2
777         read(31,*) (d(i,j),j=1,ny)
        ncol=1
c... interpolate grid points for first row
      do 2 i=1,ndely
      y=1+real(i-1)/real(ify)+eps
      ym=amod(y,1.)
```

```
      iy=int(y)
      di(i)=d(1,iy)+ym*(d(1,iy+1)-d(1,iy))+tide
      call getkcg(di(i)*dcon,rk,cg)
      ccg(i)=cg
2      kbar(2,i)=rk
c
c... figure out which bathymetry grid columns should be used
c
3      icol=1+real(lm)/real(ifx)+eps
      if(icol.ne.ncol) then
        do 5 i=1,ny
5         d(1,i)=d(2,i)
        read(31,*,end=999) (d(2,j),j=1,ny)
999        ncol=icol
      endif
      dd1=d(1,100)
c
c... bilinear interpolate depths for new column of k and cg
c
15     x=1+real(lm)/real(ifx)+eps
      xm=amod(x,1.)
      do 20 i=1,ndely
      y=1+real(i-1)/real(ify)+eps
      ym=amod(y,1.)
      iy=int(y)
      di(i)=d(1,iy)+xm*(d(2,iy)-d(1,iy))+ym*(d(1,iy+1)-d(1,iy))+
     &   xm*ym*(d(2,iy+1)-d(1,iy+1)-d(2,iy)+d(1,iy))+tide
      if(di(i).le.01) then
      di(i)=.01
      endif
c
c... shift k and ccg col 2 to 1 and calculate new 2's
c
          if(di(i).lt.dc) di(i)=dc
      kbar(1,i)=kbar(2,i)
      call getkcg(di(i)*dcon,rk,cg)
      kbar(2,i)=rk
20     ccg(i)=cg
      kbar(1,ndely+1)=kbar(1,ndely)
      kbar(2,ndely+1)=kbar(2,ndely)
      ccg(ndely+1)=ccg(ndely)
      di(ndely+1)=di(ndely)
c
```

```
      return
      end
```

For high resolution local refraction/diffraction calculations within the Torrey Pines Sub-Cell, we use the **oceanrds_tp.for** codes on a 441 x 236 raster formatted grid found in Appendix D. The input parameters output files which are required by **oceanrds_tp.for** are:

subbot50.grd...............*bathymetry input file
1.0.........................*(gis) if water values are negative gis= -1.0, if positive gis=1.0
1.........................* wave exposure 1=west, 2=north, 3=east, 4=south (icoast)
0.0.........................* sea level adjustment MSL meters (sealev) (+ = deeper water)
50.0.........................* inner grid dimensions in meters perpendicular to coast (sx)
50.0.........................* inner grid dimensions in meters parallel to coast (sy)
236.........................* number of grid cells in from deep water perpendicular to coast raster (nx)
441.........................* number of grid cells along coast from top edge (ny)
15.0.........................* wave period in seconds (persw)
285.0.........................* wave direction degress clockwise from true north (asw)
2.0.........................* wave height meters (hsw)

```
c***********************************************************************
c
c       Ocean refraction - diffraction module oceanrds_tp.for
c
c    programmed to run on gis elevation data (tp_subbot50_grd.txt subbot50.grd)
c    extracted from GIS grid  by gis_ascii_utm-xyz.for
c                          14 sep, 2004
c
c***********************************************************************
c  Bathymetry is read from
c   a formatted 441 (row) x 236 (col) real number array called 'ifile'.grd,
c   created by gis_ascii_utm-xyz.for, where'ifile' corresponds to
c   the inputted site name.  Program uses "oceanrd_tp.inp" input file.
c   All output files are named 'ifile' with different extensions,
c   ie. 'ifile'.wh1.
c***********************************************************************
c***********************************************************************
c
            parameter (max=10000000)
        character  name*8,ifile*12
        character ofile1*12,ofile2*12,ofile3*12,ofile4*12
        character ofile5*12,ofile6*12,ofile7*12
       dimension  ccg(max),di(max),hab(max),rlb(max),dib(max),ih(max)
       dimension depth(4644,4644),wht(4644,4644)
       dimension ang(4644,4644)
        dimension depthold(4644,4644)
        real kbar(2,max),kave
c
        complex aa(max),bb(max),cc(max),dd(max),uu(max),aprev(max)
        complex c3,t1,t2,t3,f,alast(max),aphys(max),mim,mip
c
        common /grid/ ny,sy,sx,dely,delx,ndely,freq,dcon,ify,ifx,tide
             common /cut/ dc
c
        pi=acos(-1.)
c
c    open parameter input file, return error message if missing
c
c
c
        open(2,file='oceanrds_tp.inp',status='old',err=900)
        read(2,'(a)') name
            read(2,*) gis
```

```
      read(2,*) icoast
         read(2,*) tide_elev
         read(2,*) sx
      read(2,*) sy
         read(2,*) nx
      read(2,*) ny
         read(2,*) persw
         read(2,*) asw
         read(2,*) hsw
c
      tide=tide_elev
      amp=hsw
      dir=asw
      per=persw
c
         freq=1/per
c
c... change to proper rotation frame, flag if theta not between +/- 45
c
      IF (icoast .EQ. 1) coast=270.0
      IF (icoast .EQ. 2) coast=90.0
      IF (icoast .EQ. 3) coast=0.0
      IF (icoast .EQ. 4) coast=180.0
c
         theta=coast-dir
c
c... set some variables to constant values
c
c       dcon = 1. for nuwave version
       dcon=1.
c
c       cutoff depth (dc) -5.0 meters
       dc=-5.0
c
c       breaking wave switch (ibreak) (1=on, 0=off)
       ibreak=1
c
c       breaking criteria (bc) 0.5  for wave height = (bc)*depth
       bc=0.5
c
c       lateral b.c.: ibc=(0) transmitting, ibc=(1) reflective
       ibc=0
c       if trans: (isn=0) straight snell, isn=(1) kirby's improved
```

```
        isn=1
c        idf=(0) small angle diff, idf=(1) large angle dif
        idf=1
c
c
           write(*,'(5(/),10x,a)')
     &'  KIRBY HIGHER ORDER REFRACTION-DIFFRACTION PROGRAM '
           write(*,'(/,10x,a)')
     &'- based on the parabolic equation method (PEM) of solving'
           write(*,'(10x,a,5(/))')
     &'  the mild-slope equation. '
c
           nn=8
           do 3 m=8,1,-1
3          if(name(m:m).eq.' ') nn=m-1
c
c
c... open grid file
c
           write(ifile,'(a,a)') name(:nn),'.grd'
       open(20,file=ifile,status='old',err=900)
c
c
c... open breaker ix,iy,wave height, wave angle, depth file
       write(ofile7,'(a,a)') name(:nn),'.bra'
        open(39,file=ofile7)
c
c
c... open sea level corrected ascii bathymetry file
       write(ofile1,'(a,a)') name(:nn),'.dep'
       open(31,file=ofile1)
c
c
c... open ascii wave number file
       write(ofile2,'(a,a)') name(:nn),'.wvn'
       open(32,file=ofile2)
c
c
c
c... open ascii wave height file
      write(ofile3,'(a,a)') name(:nn),'.wh1'
      open(33,file=ofile3)
c
```

```
c
c
c... open ascii wave angle file
      write(ofile4,'(a,a)') name(:nn),'.an1'
      open(34,file=ofile4)
c
c
c
c....  create depth array
c
      do 111 j=ny,1,-1
      read(20,*) (depthold(j,i),i=1,nx)
111    continue
      rewind(20)
c
c 10oct02 *******correct for sea level and change sign if GIS data
      do 711 i=1,nx
      do 811 j=1,ny
      depthold(j,i)=(gis*depthold(j,i))+sealev
811    continue
711    continue
c
c......write rotated ascii depth file for internal use in ref/dif
165    format(2405f12.2)
      DO 101 i=1,nx
      WRITE(31,165)(depthold(j,i),j=1,ny)
101    CONTINUE
      rewind(31)
c
c
      do 119 i=1,nx
      read(31,*) (depth(i,j),j=1,ny)
119    continue
      rewind(31)
c
c
c... read first depth to initialize offshore boundary
c
         read(31,*) (di(m),m=1,ny)
         ridep=di(ny/2)
         rewind(31)

c.... would like a grid step size on the order of 1/5 wavelength
```

```
            call getkcg(10.,wk,cg0)
            wl=2*pi/wk
            ifx=1+sx/(.2*wl)
            ify=1+sy/(.2*wl)
c
            nmx=ifx
            nmy=ify
c
        amp=amp/2.
c
c... calculate dimensions of interpolated grid
c
            dely=sy/ify
            delx=sx/ifx
        ndely=(ny-1)*ify
        ndelx=(nx-1)*ifx
c
c
        call getkcg(ridep,wk0,cg0)
c  wave length (m)
        wl=2*pi/wk0
c  wave frequency (rad/sec)
        sig=2*pi/per
c  radder's correction factor
        do 33 j=1,ndely+1
 33     ccg(j)=sqrt(sig*cg0/wk0)
c
c
             ltype=2
c... open binary wave height file
        write(ofile5,'(a,a)') name(:nn),'.bw1'
         open(9,file=ofile5,form='unformatted')
c
c... open binary wave angle file
        write(ofile6,'(a,a)') name(:nn),'.ba1'
         open(11,file=ofile6,form='unformatted')
c
c.. initialize depth array
            do 555 nn=1,ndely+1
            hab(nn)=0.
            ih(nn)=0
555     continue
```

```
c
c   enter initial condition at x=0
c
      call inbc(amp,wk0,theta,ndely,dely,alast)
c
      do 202 j=1,ndely+1
      kbar(1,j)=wk
202     kbar(2,j)=wk
c
c scale alast as in radder(1978)
      do 32 j=1,ndely+1
      alast(j)=alast(j)*ccg(j)
32      continue
c
c       call wwave(alast,ndely,nmy,dely,kbar)
c
c    solution of the parabolic eqn. by the crank-nicholson formulation
c
c    start x increments
c
      c2=1./2./dely**2
      c3=2.*(0,1)/delx
c
c---------------------------------------------------------------------
c increments in x-direction
c
      ikount=0
      do 100 l=1,ndelx
c
c... write every 10th step to the screen
c
         if(10*((l+1)/10).eq.l+1) then
      print *,' column ',l+1,' of ',ndelx+1
         endif
c
      lm=l
c
      call intkcg(lm,kbar,ccg,di,ikount)
c
c correction factor
      do 34 j=1,ndely+1
      ccg(j)=sqrt(sig*ccg(j)/kbar(2,j))
34      continue
```

```
c---------------------------------------------------------------------
c increments in y-direction - 1st. round
      do 200 j=2,ndely
c
      kave=(kbar(2,j)+kbar(1,j))/2.
c      kave=kbar(2,j)
c
      if(idf.eq.0) then
c small angle diffraction approximation
c
      t1=c2
      t3=c3*kave
      f=2*kave*(kave-wk)+c3/2*(kbar(2,j)-kbar(1,j))
c
      aa(j-1)=t1
      bb(j-1)=t3-2.*t1+f/2.
      cc(j-1)=t1
      dd(j-1)=-t1*alast(j+1)+(t3+2*t1-f/2)*alast(j)-t1*alast(j-1)
c
      endif
c
      if(idf.eq.1) then
c large angle diffraction approximation
c
      t1=c2/2.*(3.-wk/kave)
      t2=c2*c3/4./kave
      t3=c3*kave
      f=2*kave*(kave-wk)+c3/2*(kbar(2,j)-kbar(1,j))
c
      aa(j-1)=t1+t2
      bb(j-1)=t3-2.*(t1+t2)+f/2.
      cc(j-1)=t1+t2
      dd(j-1)=(t2-t1)*(alast(j+1)+alast(j-1))+(t3-2*(t2-t1)-f/2)
     &    *alast(j)
      endif
c
      if(j.eq.2) then
c      if(ibc.eq.1) cc(1)=aa(1)+cc(1)
c reflective b.c. : a1=a3
      if(ibc.eq.1) bb(1)=aa(1)+bb(1)
c reflective b.c. : a1=a2
      if(ibc.eq.0) then
c transmitting b.c.
```

```
c
      if(isn.eq.0) then
c straight snell
      mip= (0.,1.)*wk*sin(theta)/2.
      mip=(1./dely-mip)/(1./dely+mip)
      else
c  kirby's improved
      mip=(alast(2)-alast(1))/(alast(2)+alast(1))
      mip=(1.-mip)/(1.+mip)
      endif
c
      bb(1)=bb(1)+mip*aa(1)
      endif
      aa(j-1)=(0.,0.)
c
      endif
c
      if(j.eq.ndely) then
c      if(ibc.eq.1) aa(j-1)=aa(j-1)+cc(j-1)
c reflective b.c. : an=an-2
      if(ibc.eq.1) bb(j-1)=bb(j-1)+cc(j-1)
c reflective b.c. : an=an-1
      if(ibc.eq.0) then
c transmitting b.c.
c
      if(isn.eq.0) then
c   straight snell
      mim= (0.,1.)*wk*sin(theta)/2.
      mim=(1./dely+mim)/(1./dely-mim)
      else
c   kirby's improved
      mim=(alast(j+1)-alast(j))/(alast(j+1)+alast(j))
      mim=(1.+mim)/(1.-mim)
      endif
c
      bb(j-1)=bb(j-1)+mim*cc(j-1)
      endif
      cc(j-1)=(0.,0.)
      endif
c
200   continue
c----------------------------------------------------------------------
c
```

```
c                  *****   compute the solution  *****
c
        neqs=ndely-1
c
        call tridag(aa,bb,cc,dd,uu,neqs)
c
        do 143 j=2,ndely
143     alast(j)=uu(j-1)
c load alast(j)
c
        if(ibc.eq.1) then
c        alast(1)=alast(3)
c reflective b.c.: a1=a3
c        alast(ndely+1)=alast(ndely-1)
c                  : an=an-2
        alast(1)=alast(2)
c reflective b.c.: a1=a2
        alast(ndely+1)=alast(ndely)
c                  : an=an-1
        endif
c
        if(ibc.eq.0) then
c transmitting b.c.
c
        alast(1)= mip*alast(2)
        alast(ndely+1)= mim*alast(ndely)
c
        endif
c
c transform back into phys. height
        do 37 j=1,ndely+1
37      aphys(j)=2.*alast(j)/ccg(j)
c
c            *********    check for breaking    *********
c
            if(ibreak.eq.1) then
        do 54 j=1,ndely+1
        hb=bc*di(j)
        rat=hb/cabs(aphys(j))
              if(di(j).lt.0) rat=0.
        if(rat.lt.1) then
c  save point just before wave first breaks
               if(ih(j).eq.0) then
```

```
                     hab(j)=cabs(aprev(j))
                     rlb(j)=(lm-1)/real(ifx)
c  calculate direction before breaking
                      if(j.eq.ndely+1.or.ih(j+1).eq.1) then
          xx=aimag((aprev(j)-aprev(j-1))/(dely*kbar(1,j)*aprev(j)))
                       if(xx.gt.1) xx=1.
                     if(xx.lt.-1.) xx=-1.
                       dib(j)=asin(xx)
                      else
          xx=aimag((aprev(j+1)-aprev(j))/(dely*kbar(1,j)*aprev(j)))
                       if(xx.gt.1) xx=1.
                       if(xx.lt.-1.) xx=-1.
                     dib(j)=asin(xx)
                    endif
                 dib(j)=270.-dib(j)*57.296-rot
                       if(dib(j).lt.0) dib(j)=360+dib(j)
            ih(j)=1
            endif
        aphys(j)=rat*aphys(j)
        alast(j)=rat*alast(j)
          endif
54        continue
        endif
c
c  ******************************************************************
c
c... writing wave field
          if(nmx*(lm/nmx).eq.lm.or.lm.eq.1) then
         call wwave(aphys,ndely,nmy,dely,kbar)
          endif
c
          do 55 nn=1,ndely+1
          aprev(nn)=aphys(nn)
55        continue
c
100     continue
c
c... output prebreak heights and directions
          open(24,file='break.dat')
          write(24,'(/,18x,a)') '       REFRACTION - DIFFRACTION        '
          write(24,'(/,18x,a)') ' WAVE HEIGHT AND DIRECTION BEFORE BREAKING'
          write(24,'(/,15x,a,a,a,f5.1,a,f4.1)')
     &  ' site: ',name,'   direction: ',dir,'  period: ',per
```

```
        write(24,'(/,20x,a,6x,a,6x,a,9x,a,//)')
     &'row',' H (m) ','col','dir'
        m=1
        write(24,'(18x,i4,f12.2,f12.1,f12.1)') m,hab(m),rlb(m),dib(m)
      alphab=dib(m)-shor
      irlb=NINT(rlb(m))+1
      breakd=(5.0/4.0)*hab(m)
***rasterisze break point file m.....ixbra
      ixbra=2405-m
      write(39,'(18x,i4,i4,f12.3,f12.1,f12.3)') ixbra,irlb,hab(m),
     &alphab,breakd
        if(nmy.eq.1) then
        is=2
        else
        is=nmy
        endif
        mm=1
        do 888 m=is,ndely,nmy
        mm=mm+1
        write(24,'(18x,i4,f12.2,f12.1,f12.1)') mm,hab(m),rlb(m),dib(m)
      alphab=dib(m)-shor
      irlb=NINT(rlb(m))+1
      breakd=(5.0/4.0)*hab(m)
      ixbra=2405-mm
      write(39,'(18x,i4,i4,f12.3,f12.1,f12.3)')ixbra,irlb,hab(m),
     &alphab,breakd
888     continue
c
c
      rewind (9)
      rewind (11)
c
c......create wave height and wave angle arrays
      do 114 i=1,nx
          read(9) (wht(i,j),j=1,ny)
          read(11) (ang(i,j),j=1,ny)
114     continue
c
c.... write array values to:
c.... ascii depth, wave number, wave angle, wave height grid files
c
c
c2002     format(4644f12.2)
```

```
c
cccc Comment out old way of writing arrays
cccccccccc        do 134 i=1,nx
cccccccccc        write(33,2002) (wht(i,j),j=1,ny)
cccccccccc        write(34,2002) (ang(i,j),j=1,ny)
cccccccccc134     continue
c
CCCCCC Raster way of writing arrays!!!!!!!!!!!!!
      DO 176 j=ny,1,-1
        write(33,6666) (wht(i,j),i=1,nx)
        write(34,6666) (ang(i,j),i=1,nx)
  176  CONTINUE
6666    format(236f12.2)
6667    format(236f12.6)
c
c
          go to 901
900       write(*,'(20x,a,a,a)') '**** error **** ',ifile,' missing'
          write(*,'(20x,a)') '    press any key to continue '
          read(*,'(a)') idum
901     continue
       stop
       end
c
c-------------------------------------------------------------------
c
       subroutine tridag(a,b,c,r,u,n)
c
       parameter (nmax=250002)
       complex gam(nmax),a(n),b(n),c(n),r(n),u(n),bet
c
       bet=b(1)
       u(1)=r(1)/bet
       do 11 j=2,n
       gam(j)=c(j-1)/bet
       bet=b(j)-a(j)*gam(j)
       if(bet.eq.0)pause
       u(j)=(r(j)-a(j)*u(j-1))/bet
11      continue
       do 12 j=n-1,1,-1
       u(j)=u(j)-gam(j+1)*u(j+1)
12      continue
       return
```

```
      end
c
c----------------------------------------------------------------------
c
      subroutine wwave(aphys,ndely,nmy,dely,kbar)
c
c  compute & write wave height and direction, initial breaking point
c     imax= max grid size nmax=max no. of steps
c
      parameter (nmax=45000000,imax=4500000)
         dimension amod(nmax),alfa(nmax)
         real kbar(2,nmax)
         complex aphys(nmax)
c
         nky=0
      do 350 j=1,ndely+1
         if(nmy*(j/nmy).eq.j.or.j.eq.1) then
         nky=nky+1
c
          amod(nky)=cabs(aphys(j))
          if(amod(nky).gt..05) then
          if(j.eq.ndely+1) then
          xx=aimag((aphys(j)-aphys(j-1))/(dely*kbar(2,j)*aphys(j)))
          if(xx.gt.1) xx=1.
          if(xx.lt.-1.) xx=-1.
          alfa(nky)=asin(xx)
          else
          xx=aimag((aphys(j+1)-aphys(j))/(dely*kbar(2,j)*aphys(j)))
          if(xx.gt.1) xx=1.
          if(xx.lt.-1.) xx=-1.
          alfa(nky)=asin(xx)
          endif
          else
          alfa(nky)=0
          endif
          alfa(nky)=alfa(nky)*57.296
         endif
350    continue
c
       write(9) (amod(jj),jj=1,nky)
       write(11) (alfa(jj),jj=1,nky)
c
      return
```

```
      end
c
c-----------------------------------------------------------------------
c
      subroutine inbc(amp,wk,theta,ndely,dely,aa)
c
      complex aa(*)
c
      real ky
      pi=acos(-1.)
      theta=theta*pi/180.
      ky=wk*sin(theta)
c
      do 3 j=1,ndely+1
      ar=amp*cos(ky*(j-1)*dely)
      ai=amp*sin(ky*(j-1)*dely)
c      ar=(cos(ky*(j-1)*dely)+cos(-ky*(j-1)*dely))/2.
c 2 identical
c      ai=(sin(ky*(j-1)*dely)+sin(-ky*(j-1)*dely))/2.
c waves summed
      aa(j)=cmplx(ar,ai)
3      continue
c
      return
      end
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
      subroutine getkcg(d,k,cg)
c
c....  depth (meters) and frequency are input
c....  returns wave number (2pi/l) and group velocity
c
      real k
      common /grid/ ny,sy,sx,dely,delx,ndely,f,dcon,ify,ifx,tide
      data tpi/6.2831853/
c
      sig=tpi*f
      a=d*sig*sig/9.81
      if(a.ge.1) then
            yhat=a*(1+1.26*exp((-1.84)*a))
            t=exp((-2)*yhat)
            aka=a*(1+2*t*(1+t))
```

```
      else
            aka=sqrt(a)*(1+a/6.*(1+a/5.))
      endif
      k=aka/d
      x=2*k*d
      cg=tpi*(f/k)*.5*(1+x/sinh(x))
      return
      end
c
c*****************************************************************
c
      subroutine intkcg(lm,kbar,ccg,di,ikount)
c
         parameter(max=45000002)
      real kbar(2,max)
      dimension ccg(max),d(2,max),di(max)
c
      save d
           common /cut/ dc
      common /grid/ ny,sy,sx,dely,delx,ndely,f,dcon,ify,ifx,tide
      data istart,ncol,eps/1,0,.0000001/
c
c... if this is the first call, load first two columns
c
      go to (1,3) istart
1       istart=2
c... read in first 2  columns
           do 777 i=1,2
777          read(31,*) (d(i,j),j=1,ny)
        ncol=1
c... interpolate grid points for first row
      do 2 i=1,ndely
      y=1+real(i-1)/real(ify)+eps
      ym=amod(y,1.)
      iy=int(y)
      di(i)=d(1,iy)+ym*(d(1,iy+1)-d(1,iy))+tide
      call getkcg(di(i)*dcon,rk,cg)
      ccg(i)=cg
2      kbar(2,i)=rk
c
c... figure out which bathymetry grid columns should be used
c
3      icol=1+real(lm)/real(ifx)+eps
```

```
      if(icol.ne.ncol) then
        do 5 i=1,ny
5         d(1,i)=d(2,i)
        read(31,*,end=999) (d(2,j),j=1,ny)
999       ncol=icol
      endif
      dd1=d(1,100)
c
c... bilinear interpolate depths for new column of k and cg
c
15      x=1+real(lm)/real(ifx)+eps
      xm=amod(x,1.)
      do 20 i=1,ndely
      y=1+real(i-1)/real(ify)+eps
      ym=amod(y,1.)
      iy=int(y)
      di(i)=d(1,iy)+xm*(d(2,iy)-d(1,iy))+ym*(d(1,iy+1)-d(1,iy))+
     &   xm*ym*(d(2,iy+1)-d(1,iy+1)-d(2,iy)+d(1,iy))+tide
      if(di(i).le.01) then
      di(i)=.01
      endif
c
c... shift k and ccg col 2 to 1 and calculate new 2's
c
           if(di(i).lt.dc) di(i)=dc
      kbar(1,i)=kbar(2,i)
      call getkcg(di(i)*dcon,rk,cg)
      kbar(2,i)=rk
20      ccg(i)=cg
      kbar(1,ndely+1)=kbar(1,ndely)
      kbar(2,ndely+1)=kbar(2,ndely)
      ccg(ndely+1)=ccg(ndely)
      di(ndely+1)=di(ndely)
c
      return
      end
```

APPENDIX D

CODE FOR THE OCEANBAT BATHYMETRY RETRIEVAL MODULE

Appendix D:
Code for the OCEANBAT Bathymetry Retrieval Module

```
c**********************************************************************
c
c              Bathymetry data base retrieval module oceanbat.f
c                    Scott A. Jenkins & Joseph Wasyl Jan 10, 2001
c
c***********************************************************************
c
c   THE LATTITUDE AND LONGITUDE OF THE UPPER LEFT CORNER OF A
c   BATHYMETRY GRID REQUIRED.
c   (The dimensions are specified in number of 3 second lat/lon points)
c    (we have been using a default value of 200 x 200)
c
c  reads packed grid files for the Southern California Bight
c  and writes the data to an 200x200 formatted data file, 'site'.grd.
c
c***********************************************************************
      integer*2 gdata(1201,801),dlist(12,10)
      character name*8,ofile*20,ifile*60,lfile*13,cfile*13
      character name2*60
      dimension ilat(3),ilon(3)
            real*4 latmin,latmax,lonmin,lonmax
      common /param/ ixo,iyo,ix,iy,ixmin
      common /int/ dx,dy,dgx,dgy,ang,ngx,ngy
c
c... initialize the disk list
c
      call list(dlist)
c
c... get name of the active study area
     open(11,file='oceanbat.inp',status='old')
c
      read(11,'(a)') name
c
c
      read(11,'(a)') name2
      write(namesub,'(a)') name2
c
      nn=8
      do 2 m=8,1,-1
      if(name(m:m).eq.' ') nn=m-1
2      continue
c
      write(lfile,'(a,a)') name(:nn),'.log'
c
```

```
        open(8,file=lfile)
4        continue
c
        write(*,'(//,a)')
     &' OCEAN BATHYMETRY MODULE FOR THE CALIFORNIA COASTLINE
DATABASE'
c
c
c
      read(11,*) ilat(1)
      read(11,*) z
      ilat(2)=int(z)
      ilat(3)=nint(60.*(z-int(z)))
      read(11,*) ilon(1)
      read(11,*) z2
      ilon(2)=int(z2)
      ilon(3)=nint(60.*(z2-int(z2)))
c
        read(11,*) nx
        ngx=nx
c
        read(11,*) ny
         ngy=ny
c
c.....read in upper left corner as origin lat lon
c......write files across from left to right starting at origin
        nrot=3
c
c... figure out dimension of 3 sec by 3 sec grid
c
        dy=92.6
        rlat=real(ilat(1))+real(ilat(2))/60+real(ilat(3))/3600
        dx=dy*cos(rlat*.017453)
c
        if(nrot.eq.1.or.nrot.eq.3) then
          dsave=dx
          dx=dy
          dy=dsave
        endif
c
c
        imx=0
        imy=0
c
c...output information to log file
c
        write(8,'(6i4,a)') ilat,ilon
```

```
      write(8,'(2f5.1,a)') dx,dy
      write(8,'(2i5,a)') ngx,ngy
      write(8,'(a)') ' 0 0 0 0'
c
      ang=0
      ang=pi*ang/180
      if(ang.lt.0) ang=2*pi+ang
c
c... truncate origin to a grid point
c
      ilat(3)=3*(nint(real(ilat(3))/3.))
      ilon(3)=3*(nint(real(ilon(3))/3.))
c
      iyo=ilat(1)*1200+ilat(2)*20+ilat(3)/3-imy
      ixo=150000-(ilon(1)*1200+ilon(2)*20+ilon(3)/3)-imx
c
c... adjust origin based on rotation
c
      if(nrot.eq.1) then
            ixo=ixo-ngy+1
            nsave=ngy
            ngy=ngx
            ngx=nsave
       elseif(nrot.eq.2) then
            ixo=ixo-ngx+1
            iyo=iyo-ngy+1
       elseif(nrot.eq.3) then
            iyo=iyo-ngx+1
            nsave=ngx
            ngx=ngy
            ngy=nsave
       endif
c
c... find range of necesssary input data in both lat/lon and
c     database units.
c
c
      ixmin=ixo
      ixmax=ixo+ngx-1
      iymin=iyo
      iymax=iyo+ngy-1
c
c
c... convert back to lat/lon coordinates
c
      latmin=real(iymin-1)/1200.
      latmax=real(iymax)/1200.
```

```
      lonmax=(150000.-(ixmin+1))/1200.
      lonmin=(150000.-ixmax)/1200.
c
      print *,' lat/lon range ',latmin,latmax,lonmin,lonmax
c
      do 60 i=latmin,latmax
      nggx=1
      nggy=1
          iy=i*1200
          nys=iymin-iy
          nye=iymax-iy
          if(nys.lt.1) then
           nggy=abs(nys)+2
           nys=1
          endif
          if(nye.gt.1200) then
          if(nye.gt.2400) then
           nye=1200
           else
           nye=1200
           nggy=1
           endif
          endif
      do 60 j=lonmax,lonmin,-1
          ix=150000-(j+1)*1200-1
          nxs=ixmin-ix
          nxe=ixmax-ix
          if(nxs.lt.1) then
           nggx=abs(nxs)+2
           nxs=1
          endif
          if(nxe.gt.1200) then
c          if(nxe.gt.2400) then
c               nxe=1200
c               nggx=1201
c               else
c           nggx=1
           nxe=1200
           endif
c          endif
c
c... figure out which bathymetry file to read
c
      il=i-31
      jl=j-116
      ndisk=dlist(il,jl)
c
```

```
 45    continue
c
c
      write(ifile,1000)name2(1:LSTGDCHR(name2)), i,j
c
1000   format(a,'/',i2,i3,'pg.dta')
c
      close(20)
      print *,' searching: ',ifile
      open(20,file=ifile,status='old',form='unformatted',err=50)
      go to 51
50     write(*,'(a,a,a)')
     +  ' ????? unable to find    ',ifile,'  on the disk ????? '
      write(*,'(a,\)') ' abort loading ? (y/n) : '
      read(*,*) ans
      if(ans.eq.'y'.or.ans.eq.'Y') then
      stop
      else
      go to 45
      endif
c
51     continue
c
      call unpack(gdata,nxs,nxe,nys,nye,nggx,nggy)
c
60     continue
c
c... output the grid array, one s to n column at a time from w to e
c
      write(ofile,2000) name(:nn)
2000   format(a,'.grd')
c
      open(12,file=ofile)
c
c... output depends on rotation
c
2009   format(200f8.3)
      if(nrot.eq.1) then
      do 70 j=1,ngy
      write(12,2009) (real(gdata(i,j))*.1+.853,i=ngx,1,-1)
70     continue
      elseif(nrot.eq.2) then
      do 71 i=ngx,1,-1
      write(12,2009) (real(gdata(i,j))*.1+.853,j=ngy,1,-1)
71     continue
      elseif(nrot.eq.3) then
      do 72 j=ngy,1,-1
```

```
      write(12,2009) (real(gdata(i,j))*.1+.853,i=1,ngx)
72     continue
      else
      do 73 i=1,ngx
      write(12,2009) (real(gdata(i,j))*.1+.853,j=1,ngy)
73     continue
      endif
c
c... make the default contour commmand file for the base map
c
c... maximum length is 5.5 inches
c
      rlmax=max(real(ngx)*dx,real(ngy)*dy)
      h=6.5*real(ngy)*dy/rlmax
      w=6.5*real(ngx)*dx/rlmax
      write(cfile,'(a,a)') name(:nn),'.cnt'
      open(7,file=cfile)
      write(7,'(a,a)') 'file     ',ofile
      write(7,'(a,a)') 'format   binary'
      write(7,*) 'axes ',lonmax,lonmin,latmin,latmax
      write(7,'(a)') 'xlabel Latitude'
      write(7,'(a)') 'ylabel Longitude'
      write(7,'(a)') 'caption'
      write(7,'(a)') 'title  Depth Contours (meters from MSL)'
      write(7,'(a)') 'letter   .1 .1 .1 .1 .1'
      write(7,'(a,f5.2)') 'width    ',w
      write(7,'(a,f5.2)') 'height   ',h
      write(7,'(a,i4,i4)') 'read     ',ngx,ngy
      write(7,*) 'plot ',.5*(8.5-w),.5*(11.-h)
      write(7,'(a)') 'stop'
c
c.......................................................................
c.........write ascii bathymetry
c        rewind(12)
c
c
c... open ascii bathymetry file
c       write(ofile1,'(a,a)') name(:nn),'.dep'
c       open(17,file=ofile1)
c
c
      end
c
c
c*******************************************************************
c
      subroutine unpack(gdata,nxs,nxe,ns,ne,ngx,ngy)
```

```
c
c... longitudes are reversed as read in to array since opposite of xy coor.
c
      integer*2 gdata(1201,801),dum
      integer*2 rec(1200),pdata(1200),idata(1200),dmin,dmax
      common /param/ ixo,iyo,ix,iy,ixmin
      data dmin,dmax/5001,-999/
c
c
c... loop through the files with desired data
c
      read(20) rec
c
c...  read records
c
      ny=ngy
      do 100 i=1,ne
c      print *, ' reading rec ',i
c...  unpacking data
c
c... check if record only contains deep water or land
c
      if(rec(i).lt.0) then
      if(i.lt.ns) go to 100
      if(rec(i).eq.-1) then
            depth=3050
      else
            depth=-20
      endif
            do 33 m=1,1200
33            idata(m)=depth
            go to 91
      endif
c
c... unpack the row
c
      icnt=1201
c
      if(i.lt.ns) then
      read(20) dum
      go to 100
      endif
c
      read(20) (pdata(m),m=1,rec(i))
c

      do 90 j=1,rec(i)
```

```
c
      if(pdata(j).lt.-10000) then
            if(pdata(j).lt.-20000) then
                  ncnt=abs(pdata(j)+20000)
                  depth=-20
            else
                  ncnt=abs(pdata(j)+10000)
                  depth=3050
            endif
            do 85 m=1,ncnt
            icnt=icnt-1
85            idata(icnt)=depth
            go to 90
      endif
c
      icnt=icnt-1
      idata(icnt)=pdata(j)
c
90     continue
c
c... place desired data from record into grid array
c
91     continue
      nx=ngx
c
      do 92 jj=nxs,nxe
c
      gdata(nx,ny)=idata(jj)
c      dmax=max0(gdata(kx,ky),dmax)
c       dmin=min0(gdata(kx,ky),dmin)
 92    nx=nx+1
c
 98    ny=ny+1
c
c.... save rec
c
100    continue
c      print *,dmin,dmax
c
c      ngx=nx
c      ngy=ny
      return
      end
c
c
c*****************************************************************************
c
```

```
      subroutine list(dlist)
c
c... contains the floppy disk no. for grid files i=lat-31; j=lon-116
      integer*2 dlist(12,10)
c
      do 10 i=1,12
      do 10 j=1,10
      dlist(i,j)=-1
10    continue
c
      dlist(1,1)=1
      dlist(1,2)=1
      dlist(1,3)=1
      dlist(2,2)=1
      dlist(2,1)=2
      dlist(2,3)=2
      dlist(3,2)=2
      dlist(4,4)=2
      dlist(2,4)=3
      dlist(5,5)=3
      dlist(3,3)=3
      dlist(3,4)=4
      dlist(4,5)=4
      dlist(5,6)=5
c
      return
      end
c***********************************************************************
      INTEGER FUNCTION LSTGDCHR(NAMESUB)

      CHARACTER*(*)     NAMESUB
      INTEGER     II,CLEN,INDEX

      CLEN = LEN(NAMESUB)
      DO 100 II = CLEN,1,-1
100   IF( ICHAR(NAMESUB(II:II)) .GT. 60 .AND.
     1     ICHAR(NAMESUB(II:II)) .LT. 127 )GOTO 101
101   INDEX=II

      LSTGDCHR = INDEX
      RETURN

      END
```

APPENDIX E

CODE FOR THE SEDXPORT TRANSPORT MODEL

Appendix E:
Code for the SEDXPORT Transport Model

```
c***********************************************************************
c
c  sedxport3.for Transport Model - (bottom, surfzone, outfall sources)
c        with NULL POINT January. 24, 2005 & selected grid point - 10:00
c          TIME STEP MODE beach outfall source and canyon sink modules
c            Littoral Advection/Diffusion
c             Fine  Sediment  Dispersion MODEL
c             written by Scott A.Jenkins & Joseph Wasyl
c
c
c***********************************************************************
c
c***********************************************************************
c
       parameter (ni=200, nj=200)
c
        real*4 n0,n0_l
c
       character name*8,infil1*12,infil2*12
       character infile3*12,ifile2*12
       Dimension wnum(ni,nj),wht(ni,nj),depth(ni,nj)
       dimension iyb(nj),ixb(nj),bh(nj),ba(nj),bd(nj)
        dimension ubr(nj),dxbr(nj),dxbave(nj),ubave(nj)
       dimension zw(5)
        dimension r7(10,5),revr7(10,5)
        dimension d(10),rcp(10),rc(10),rcsp(10),rszcp(10)
        dimension rrs(10),rrsp(10)
c
c********************null point arrays********************
        dimension type(ni,nj)
c***************************************************
c ... second outfall arrays
        dimension rrs2(10),rrsp2(10)
       dimension rrivr2(10,5),pnrr2(10,5)
c***************************************************
c
c
        dimension rsurf(10,5),rriv(10,5),rbot(10,5)
        dimension pns(10,5),pnr(10,5),pnb(10,5),pnt(10,5)
        dimension sp(10),sp2(10),spp(10)
```

```
c
         dimension suml1(ni,nj),suml2(ni,nj)
         dimension suml3(ni,nj),suml4(ni,nj),b(ni,nj),b2(ni,nj)
         dimension ux(ni,nj),uy(ni,nj)
c
c____________________________________________________________________
c........arrays unique to selected grid point loop
         dimension sum2(101),zw3(101),dep2(101),slope(101),a4402(101)
         dimension r72(10,101),revr72(10,101),dep(100)
         dimension rsurf2(10,101),rriv2(10,101),rbot2(10,101)
         dimension pns2(10,101),pnr2(10,101),pnb2(10,101),pnt2(10,101)
c
c****************************************************
c ... second outfall arrays unique to selected grid point
        dimension rriv2r2(10,101),pnr2r2(10,101)
c****************************************************
c****************************************************
c..Salinity module arrays
        dimension salmean(5),sal2mean(101)
        dimension sriv1(5),sriv2(5),salr7(5)
      dimension s2riv1(101),s2riv2(101),sal2r7(101)
      dimension sall1(ni,nj),sall2(ni,nj),sall3(ni,nj)
      dimension sall4(ni,nj)
      dimension acdom1(ni,nj),acdom2(ni,nj),acdom3(ni,nj)
      dimension acdom4(ni,nj),a440(101)
c******************************************************************
c
C
c
c
c
        open(18,file='CEM_sedxport3.inp',status='old')
c
        read(18,'(a)') name
           read(18,*) nx
        read(18,*) ny
        read(18,*) igrdx
           read(18,*) igrdy
           read(18,*) sx
        read(18,*) sy
        read(18,*) persw
        read(18,*) perwin
        read(18,*) hsw
```

```
read(18,*) hwin
read(18,*) tanbeta
read(18,*) aks
read(18,*) akb
read(18,*) ak2
   read(18,*) rci
   read(18,*) rszi
read(18,*) ak
read(18,*) q
read(18,*) irx
read(18,*) iry
read(18,*) dr
read(18,*) rwidth
read(18,*) rrsi
read(18,*) q2
read(18,*) irx2
read(18,*) iry2
read(18,*) dr2
read(18,*) rwidth2
read(18,*) rrsi2
read(18,*) ak3
read(18,*) ak4
read(18,*) ak5
read(18,*) ibins
read(18,*) verdat
read(18,*) numlay
read(18,*) alay1
read(18,*) alay2
read(18,*) alay3
read(18,*) alay4
read(18,*) surf
read(18,*) river
read(18,*) river2
read(18,*) bottom
read(18,*) rhos
read(18,*) rhosr
read(18,*) rhosr2
read(18,*) tcon
read(18,*) tconriv
read(18,*) deltat
read(18,*) timestep
read(18,*) n0
read(18,*) gamma
```

```
      read(18,*) salo
      read(18,*) w0sal
      read(18,*) ak3s
      read(18,*) ak5s
      read(18,*) aksal
      read(18,*) domslope
      read(18,*) dominter
      read(18,*) domback
      read(18,*) tconsal
      read(18,*) ak7
      read(18,*) ilat1
      read(18,*) z
      read(18,*) two_layer
      read(18,*) v
      read(18,*) ak8
      read(18,*) p_mbar
      read(18,*) dmix_for
      read(18,*) dmix
      read(18,*) ak2_l
      read(18,*) ak_l
      read(18,*) ak3_l
      read(18,*) ak5_l
      read(18,*) tcon_l
      read(18,*) tconriv_l
      read(18,*) n0_l
      read(18,*) gamma_l
      read(18,*) delsal_l
      read(18,*) w0sal_l
      read(18,*) ak3s_l
      read(18,*) ak5s_l
      read(18,*) aksal_l
      read(18,*) tconsal_l
c
c...convert time to seconds
      time=deltat*60*60
c........convert tcon by multiplying by 10 E-6
      tcon=tcon*0.001
c
      if(timestep.NE.0.0)then
      open(27,file='dmix_wind.dat',status='old')
      read(27,*) dmix_old
      read(27,*) epsilonw_old
      dmix_old=dmix_old
```

```
      close(27)
      else
c........when no forecast available dmix_old is set to nowcast n0_l at t=0
      dmix_old=n0_l
      endif
c
c.......convert background in mg/l to number of grains
      n0=n0*100
c
c
      ilat2=int(z)
      ilat3=nint(60.*(z-int(z)))
      theta=real(ilat1)+real(ilat2)/60+real(ilat3)/3600
      v=v*0.5148*100.0
      ak8=0.00009/((v**0.5)+1.1103)
      ustar=(ak8*(v**2))**0.5
      rot_e=0.000072685
      rho_a=0.001293*(p_mbar/760)**0.714285714
      f=2.0*rot_e*SIND(theta)
c
c
c......calculate for residual from previous time step
c.......epsilonw is in cgs units
      epsilonw=0.0000043*(v**2)
      if(v.LE.600)epsilonw=0.0000000102*(v**3)
      if(timestep.EQ.0.0)epsilonw_old=epsilonw
c
c..calculate sediment layer depth in meters when dmix_for=0 (no forecast)
      if(dmix_for.EQ.0.0)then
      dmix=0.4*ustar/(f*100)
      endif
c
c...if a forecast dmix is used do not modify from previous timestep
      if(dmix_for.NE.0)go to 1867
      dmix=dmix+((dmix_old-dmix)*
     &EXP(-1.0*epsilonw_old*time/(dmix_old*100.0)**2))
1867  continue
c
c............in a single layer system dmix is set to 500 meters!!
      if(two_layer.EQ.0.0)dmix=500.0
c
c
c........dmix still in meters prior to writing
```

```
         open(27,file='dmix_wind.dat',status='unknown')
         write(27,8030)dmix
         write(27,8030)epsilonw
8030     format(f20.10)
c
c........convert dmix back to centimeters for subsequent calculations
         dmix=dmix*100.0
c
        alay3=(dmix-100)
        alay4=(dmix+100)
c
c
        open(19,file='bulk_density.dat',status='old')
        read(19,*) rci1
        read(19,*) rci2
        read(19,*) rci3
        read(19,*) rci4
        read(19,*) rci5
        read(19,*) rci6
        read(19,*) rci7
        read(19,*) rci8
        read(19,*) rci9
        read(19,*) rci10
        read(19,*) rci11
        read(19,*) rci12
        read(19,*) rci13
        read(19,*) rci14
c
        ires=101
        riv2=river2
        riv=river
        bot=bottom
c
c
        salmax=(domback-dominter)/domslope*(-1.0)
c
c....divide n0 grains among the first 5 grain size bins
       an0b1=n0*.767
       an0b2=n0*.179
       an0b3=n0*.0376
       an0b4=n0*.00886
       an0b5=n0*.00609
       an0b6=n0*.00142
```

```
       an0b7=n0*.000248
       an0b8=n0*.000121
       an0b9=n0*.0000484
       if(an0b1.LE.2.0)an0b1=2.0
       if(an0b2.LE.2.0)an0b2=2.0
       if(an0b3.LE.2.0)an0b3=2.0
       if(an0b4.LE.2.0)an0b4=2.0
       if(an0b5.LE.2.0)an0b5=2.0
       if(an0b6.LE.2.0)an0b6=2.0
       if(an0b7.LE.2.0)an0b7=2.0
       if(an0b8.LE.2.0)an0b8=2.0
       if(an0b9.LE.2.0)an0b9=2.0
c
c
         per=persw
c
        numlay=4
        if(numlay.GE.4)numlay=4
c

c.....open array of bulk density values from previous time step
c
c
        OPEN(2, FILE='timestep.sed', ACCESS='DIRECT', RECL=24,
     &  FORM='UNFORMATTED')
c
c.......open planview of grain number output files
c
c
c... open ascii file layer 1 grain number / cc
        open(41,file='layer1.sed')
        open(81,file='layer1.sal')
        open(85,file='layer1.dom')
        open(91,file='layer1.slp')
c
c
c... open ascii file layer 1 grain number / cc
        open(42,file='layer2.sed')
        open(82,file='layer2.sal')
        open(86,file='layer2.dom')
        open(92,file='layer2.slp')
c
        if(two_layer.NE.0)then
```

```
c... open ascii file layer 1 grain number / cc
      open(43,file='layer3.sed')
      open(83,file='layer3.sal')
      open(87,file='layer3.dom')
c
c... open ascii file layer 1 grain number / cc
      open(44,file='layer4.sed')
      open(84,file='layer4.sal')
      open(88,file='layer4.dom')
c
      endif
c
c
c
         freq=1/per
      pi=acos(-1.)
      g=980.0
      ro=1.03
     omega=2*pi/per
c
c
         nn=8
         do 3 m=8,1,-1
3        if(name(m:m).eq.' ') nn=m-1
c
c
c... open grain size and mass percent file 'masstotal.dat'
c
      open(26,file='masstotal.dat',status='old')
c
c... read data from file
c
      nfact=1
      do 44 i=1,ibins
      read(26,*)d(i),rcp(i),rcsp(i),rrsp(i),rrsp2(i)
c ..... convert microns to centimeters
      d(i)=d(i)*.0001
      rc(i)=rcp(i)*rci
      rszcp(i)=rcsp(i)*rszi
      rrs(i)=rrsp(i)*rrsi
      rrs2(i)=rrsp2(i)*rrsi2
      nfact=nfact*i
44    continue
```

```
c
c...open selected point file at selected grid point(igrdx,igrdy)
      open(99,file='profile.sed',status='unknown')
c
c
c********************NULL POINT***************************
c... open bottom type classification file (from grass)
      open(29,file='bottom_type.dat',status='old')
c
c
c
c...open bottom distribution file at selected crossection(igrdy)
c.........this part of program commented out for integrated cwc model
c....  open(50,file='nullpt.dat',status='unknown')
c

c.....open bottom distribution file at selected point(igrdx,igrdy)
c.......this part commented out for integrated model
c.     open(88,file='nullpt.igrdx')
c
c**********************************************************
c
c... open ascii wave number file from swell component
      write(ifile2,'(a,a)') name(:nn),'.wvn'
      open(32,file=ifile2,status='old')
c
c... open ascii depth file corrected to tide
         write(infil1,'(a,a)') name(:nn),'.dep'
      open(20,file=infil1,status='old')
c
c.......open total wave height file
         write(infil2,'(a,a)') name(:nn),'.wht'
      open(9,file=infil2,status='old')
c
c... open ascii file x-component of current
      open(45,file='xcur.xpt')
c
c... open ascii file y-component of current
      open(46,file='ycur.xpt')
c
c
c... read bathymetry, wave height, swell wave number x1, y=1,200
c....read x-component of current, y-component of current
```

```
c.....x1, y1 is bottom left corner looking at page by convention
c
c.......read into arrays depth, wave height, wave number
        do 111 i=1,nx
        read(20,*) (depth(i,j),j=1,ny)
        read(9,*)(wht(i,j),j=1,ny)
        read(32,*)(wnum(i,j),j=1,ny)
111     continue
c
c.......read in xcur, ycur, bot type verticle system reversed (raster)
        do 168 j=ny,1,-1
        read(45,*)(ux(i,j),i=1,nx)
        read(46,*)(uy(i,j),i=1,nx)
        read(29,*) (type(i,j),i=1,nx)
c
168     continue
c
c
c..... open breaker input file
            write(infile3,'(a,a)') name(:nn),'.bra'
        open(38,file=infile3)
c
c.... read x,y, breaker height, breaker angle, breaker depth
        do 150 n=1,ny
        read(38,*) iyb(n),ixb(n),bh(n),ba(n),bd(n)
c
c..........determine wave height, wnum at the break point
         ibpx=ixb(n)-1
        whtbc=wht(ibpx,n)*100
       wnumbc=wnum(ibpx,n)/100
       depthbc=depth(ibpx,n)*100
        wndepb=wnumbc*depthbc
c
c...create ix by 200 arrays of dxbr's and ubr's at the break point
        dxbr(n)=whtbc/SINH(wndepb)
        ubr(n)=dxbr(n)*omega
c
150     continue
c
c...........calculate average values dxbave and ubave
         do 151 i=1,200
c
        im1=i-1
```

```
      im2=i-2
      ip1=i+1
      ip2=i+2
      if(im1.LT.1)im1=1
      if(im2.LT.1)im2=1
      if(ip1.GT.200)ip1=200
      if(ip2.GT.200)ip2=200
      dxbave(i)=(dxbr(i)+dxbr(im1)+dxbr(im2)+dxbr(ip1)+dxbr(ip2))/5
      ubave(i)=(ubr(i)+ubr(im1)+ubr(im2)+ubr(ip1)+ubr(ip2))/5
c
151   continue
c
c.............initilize icount to 0 outside of iy,ix, and ipart loops
c..........and rtold,rsold,qsold,qs2old,salold,salold2 for timestep=0
       icount=0
       rtold=0.0
       rsold=0.0
       qsold=0.0
       qs2old=0.0
       salold=0.0
       salold2=0.0
c
c..........determine wave height, wnum at the outfall
c       whtrm=wht(irx,iry)*100
c      wnumrm=wnum(irx,iry)/100
c      depthrm=depth(irx,iry)*100
c       wndeprm=wnumrm*depthrm
c
c
c........ at the outfall
c       drm=whtrm/SINH(wndeprm)
c       urm=drm*omega
c
      do 1800 iy=1,ny
      allxbrk=0
c
      bd1=bd(iy)*100
      if(bd1.LT.10.0)bd1=10.0
      ba1=ba(iy)
      bh1=bh(iy)*100
c
c..........determine wave height, wnum at the break point
       ibpx=ixb(iy)-1
```

```
        whtbc=wht(ibpx,iy)*100
       wnumbc=wnum(ibpx,iy)/100
       depthbc=depth(ibpx,iy)*100
        wndepb=wnumbc*depthbc
c
c........ at the break point
        dxb=whtbc/SINH(wndepb)
        ub=dxb*omega
c
c
c
         stopdep=bd(iy)
c
        do 1801 ix=1,nx
c
c***************null point - bottom type******************************
        if(type(ix,iy).LE.100)go to 7006
c
        if(type(ix,iy).LE.200)then
        rci=(rci2-rci1)*(type(ix,iy)-100)/100+rci1
        go to 7007
        endif
c
        if(type(ix,iy).LE.300)then
        rci=(rci3-rci2)*(type(ix,iy)-200)/100+rci2
        go to 7007
        endif
c
        if(type(ix,iy).LE.400)then
        rci=(rci4-rci3)*(type(ix,iy)-300)/100+rci3
        go to 7007
        endif
c
        if(type(ix,iy).LE.500)then
        rci=(rci5-rci4)*(type(ix,iy)-400)/100+rci4
        go to 7007
        endif
c
        if(type(ix,iy).LE.200)then
        rci=(rci2-rci1)*(type(ix,iy)-100)/100+rci1
        go to 7007
        endif
c
```

```
      if(type(ix,iy).LE.300)then
      rci=(rci3-rci2)*(type(ix,iy)-200)/100+rci2
      go to 7007
      endif
c
      if(type(ix,iy).LE.400)then
      rci=(rci4-rci3)*(type(ix,iy)-300)/100+rci3
      go to 7007
      endif
c
      if(type(ix,iy).LE.500)then
      rci=(rci5-rci4)*(type(ix,iy)-400)/100+rci4
      go to 7007
      endif
c
      if(type(ix,iy).LE.600)then
      rci=(rci6-rci5)*(type(ix,iy)-500)/100+rci5
      go to 7007
      endif
c
      if(type(ix,iy).LE.700)then
      rci=(rci7-rci6)*(type(ix,iy)-600)/100+rci6
      go to 7007
      endif
c
      if(type(ix,iy).LE.800)then
      rci=(rci8-rci7)*(type(ix,iy)-700)/100+rci7
      go to 7007
      endif
c
      if(type(ix,iy).LE.900)then
      rci=(rci9-rci8)*(type(ix,iy)-800)/100+rci8
      go to 7007
      endif
c
      if(type(ix,iy).LE.1000)then
      rci=(rci10-rci9)*(type(ix,iy)-900)/100+rci9
      go to 7007
      endif
c
      if(type(ix,iy).LE.1100)then
      rci=(rci11-rci10)*(type(ix,iy)-1000)/100+rci10
      go to 7007
```

```
      endif
c
      if(type(ix,iy).LE.1200)then
      rci=(rci12-rci11)*(type(ix,iy)-1100)/100+rci11
      go to 7007
      endif
c
      if(type(ix,iy).LE.1300)then
      rci=(rci13-rci12)*(type(ix,iy)-1200)/100+rci12
      go to 7007
      endif
c
      if(type(ix,iy).LE.1400)then
      rci=(rci14-rci13)*(type(ix,iy)-1300)/100+rci13
      go to 7007
      endif
c
      if(type(ix,iy).GT.1400)then
      rci=rci14
      go to 7007
      endif
c
7006  rci=rci1
7007  continue
c
       distbp=((201-ix)-ixb(iy))*sx*100
c**********************************************************
c
      uy1=uy(ix,iy)+0.000001
      ux1=ux(ix,iy)+0.000001
c
c
c
c.....ADD resident sediment volume
c........ initialize total grain variable at each x,y location
      suml1(ix,iy)=n0
      suml2(ix,iy)=n0
      suml3(ix,iy)=n0
      suml4(ix,iy)=n0
c
      sall1(ix,iy)=salo
      sall2(ix,iy)=salo
      sall3(ix,iy)=salo
```

```
        sall4(ix,iy)=salo
c
        b(ix,iy)=gamma
        b2(ix,iy)=gamma
c
c
       if(depth(ix,iy).LE.stopdep)then
       allxbrk=1.0
        do 478 irt=ix,nx
        suml1(irt,iy)=-1.0
        suml2(irt,iy)=-1.0
        suml3(irt,iy)=-1.0
        suml4(irt,iy)=-1.0
c
        sall1(irt,iy)=-1.0
        sall2(irt,iy)=-1.0
        sall3(irt,iy)=-1.0
        sall4(irt,iy)=-1.0
c
        acdom1(irt,iy)=-1.0
        acdom2(irt,iy)=-1.0
        acdom3(irt,iy)=-1.0
        acdom4(irt,iy)=-1.0
c
c
        b(irt,iy)=-1.0
        b2(irt,iy)=-1.0

478     continue
        go to 479
        endif
c
c
c
c .....  convert wave hieght, water depth, and wave number into cgs
        whtc=wht(ix,iy)*100
        wnumc=wnum(ix,iy)/100
        depthc=depth(ix,iy)*100
       wndep=wnumc*depthc
c
        d0=whtc/SINH(wndep)
        u=omega*d0
c
```

```
c
c
c
      do 1776 ipart=1,ibins
c
      dip=d(ipart)
      rrsfix2=rrs2(ipart)
      rrsfix=rrs(ipart)
      rcspfix=rszcp(ipart)
c
c************************null point begin***************************
c
      d0np=d0
      dm=ak7*d0np
      unp=rszi/(rhos*pi*dm)
      unpd1=unp*d(1)
      if(unpd1.GT.ibins)unp=ibins/d(1)
      if(unp.LT.ibins/(100*d(1)))unp=ibins/(100*d(1))
      unpdip=unp*dip
      rc(ipart)=rci*6.092*(((unp*dip)**ibins)/nfact)*EXP(-1*unp*dip)
      if(rc(ipart).LT.0.0000000000000001)rc(ipart)=0.0000000000000001
c
c............nullpt selected grid write commented out
c       if(iy.NE.igrdy)go to 8001
c       if(ix.NE.igrdx)go to 8002
c       write(88,8003)dip,rc(ipart),depth(ix,iy),unpdip
c8003   format(4e10.3)
c8002   continue
c8001   continue
c**********************null point end*******************************
      rcfix=rc(ipart)
c
c
       icount=icount+1
c
      if(timestep.EQ.0.0)go to 81
      read(2, REC=icount)rtold,rsold,qsold,qs2old,salold,salold2
81     continue
c
      if(ipart.EQ.1)w0=.00011
      if(ipart.EQ.2)w0=.00043
      if(ipart.EQ.3)w0=.0017
      if(ipart.EQ.4)w0=.0050
```

```
      if(ipart.EQ.5)w0=.0108
      if(ipart.EQ.6)w0=.044
      if(ipart.EQ.7)w0=.17
      if(ipart.EQ.8)w0=.38
      if(ipart.EQ.9)w0=.98
c
c****************************************************************
      if(iy.NE.igrdy)go to 4000
      if(ix.NE.igrdx)go to 4001
c
      ipart2=ipart
c
c
      fw=EXP(5.2*((2.5*dip/d0)**0.2)-6.0)
c
      delta=0.72*d0*(2.5*dip/d0)**0.25
      deltab=0.72*dxb*(2.5*dip/dxb)**0.25
      deltar=0.72*dxbave(iy)*(2.5*dip/dxbave(iy))**0.25
      deltas=0.72*dxbave(iy)*(0.000006625/dxbave(iy))**0.25
c
c
c......for the outfall sediment flux use running mean values of ub *dxb
     shieldr=ubave(iy)**2/((rhos/ro-1.0)*g*dip)
     epsilonr_l=0.00035*ak3_l*((ubave(iy)/w0)**0.68)*
    &(shieldr**0.4)*deltar*g*per
     if(epsilonr_l.LT.0.0000000001)epsilonr_l = 0.0000000001
     epsilonr=epsilonr_l+epsilonw
c**
c......for the outfall plume salinity use running mean values of ub *dxb
     shieldrs=ubave(iy)**2/(.03*g*dr)
     epsilonrs_l=0.00035*ak3s*((ubave(iy)/w0sal)**0.68)*
    &(shieldrs**0.4)*deltas*g*per
     if(epsilonrs_l.LT.0.0000000001)epsilonrs_l = 0.0000000001
     epsilonrs=epsilonrs_l+epsilonw
c
c.....at the local break point
      shieldb=ub**2/((rhos/ro-1.0)*g*dip)
      epsilonb_l=0.00035*ak3*((ub/w0)**0.68)*(shieldb**0.4)
    &*deltab*g*per
      if(epsilonb_l.EQ.0)epsilonb_l=0.0000000001
      epsilonb=epsilonb_l+epsilonw
c
c.....at the local break point for disequilibrium profile
```

```
      epsilonbs_l=0.00035*ak3s*((ub/w0sal)**0.68)*(shieldrs**0.4)
     &*deltas*g*per
      if(epsilonbs_l.EQ.0.0000000001)epsilonbs_l = 0.0000000001
      epsilonbs=epsilonbs_l+epsilonw
c
c.......offshore
      shield=u**2/((rhos/ro-1.0)*g*dip)
      epsilon_l=0.00035*ak2*((u/w0)**0.68)*(shield**0.4)*delta*g*per
      if(epsilon_l.EQ.0.0000000001)epsilon_l = 0.0000000001
      epsilon=epsilon_l+epsilonw
c
c......outfall shields and epsilon calculated in RIVERM subroutine
c**
c**
      rrsfix2=rrs2(ipart2)
      rrsfix=rrs(ipart2)
      rcspfix=rszcp(ipart2)
      rcfix=rc(ipart2)
c
      if(riv.EQ.1)then
      CALL RIVERM(epsilonr,dr,rwidth,q,ux1,uy1,ix,iy,irx,iry,sx,
     &sy,rrsfix,ak5,w0,deltab,rhosr,ro,dip,g,per,ak3,time,qsold,
     &depthc,tconriv,qs)
      CALL SALM1(epsilonrs,dr,rwidth,q,ux1,uy1,ix,iy,irx,iry,sx,
     &sy,salo,ak5s,w0sal,deltas,rhosr,ro,dip,g,per,ak3,time,salold,
     &depthc,tconsal,sal)
c
      else
      continue
      endif
c
      if(riv2.EQ.1)then
      CALL RIVER2M(epsilonr,dr2,rwidth2,q2,ux1,uy1,ix,iy,irx2,iry2,sx,
     &sy,rrsfix2,ak5,w0,deltab,rhosr2,ro,dip,g,per,ak3,time,qs2old,
     &depthc,tconriv,qs2)
c**
c**
      CALL SALM2(epsilonrs,dr2,rwidth2,q2,ux1,uy1,ix,iy,irx2,iry2,sx,
     &sy,salo,ak5s,w0sal,deltas,rhosr2,ro,dip,g,per,ak3,time,salold2,
     &depthc,tconsal,sal2)
c
      else
      continue
```

```
      endif
c
      if(surf.EQ.1)then
      CALL SURFLM(epsilonb,bd1,ba1,bh1,ibpx,ix,iy,
     &ak4,aks,akb,dip,w0,pi,g,rhos,ro,tanbeta,rcspfix,time,rsold,
     &depthc,tcon,rs)
      else
      continue
      endif
c
c
      if(bot.EQ.1)then
c**
c**
      CALL BOTTOMM(ix,iy,u,shield,fw,ak,pi,wndep,rcfix,time,rtold,
     &depthc,tcon,rt,w0,epsilon)
      else
      continue
      endif
c
      do 2002 k2=1,ires
      zw3(k2)=depthc/100*(k2-1)
      dep2(k2)=depthc/100*(ires-k2)
      if(k2.EQ.1)zw3(k2)=delta
      if(k2.EQ.1)dep2(k2)=depthc-delta
c
c ....sediment transport calculations
c......initialize sal2mean(k2)
     sal2mean(k2)=salo
      a4402(k2)=dominter-domslope*sal2mean(k2)
      if(sal2mean(k2).GE.salmax)a4402(k2)=domback
       if(dmix.GE.dep2(k2))then
      sal2r7(k2)=(-1.0)*dep2(k2)*aksal*w0sal/epsilonbs
      else
      sal2r7(k2)=(-1.0)*dep2(k2)*aksal*((dimx/4.21)**2.4)
     &*w0sal/epsilonbs_l
      endif
c
       if((depthc-dmix).GE.zw3(k2))then
      r72(ipart2,k2)=(-1.0)*zw3(k2)*w0/epsilon_l
      else
      r72(ipart2,k2)=(-1.0)*zw3(k2)*w0*(dmix/dip)**2.4/epsilon
      endif
```

```
         if((depthc-dmix).LT.0.0)r72(ipart2,k2)=(-1.0)
     &*zw3(k2)*w0/epsilon
c
         if(dmix.GE.dep2(k2))then
        revr72(ipart2,k2)=(-1.0)*dep2(k2)*w0/epsilonb
        else
        revr72(ipart2,k2)=(-1.0)*dep2(k2)*w0*
     &(dmix/dip)**2.4/epsilonb_l
      endif
c**
c**
        if(surf.EQ.1)then
        rsurf2(ipart2,k2)=rs*EXP(r72(ipart2,k2))
        pns2(ipart2,k2)=(6.0/(pi*dip**3))*rsurf2(ipart2,k2)/rhos
        else
        pns2(ipart2,k2)=0
        endif
C
        if(riv.EQ.1)then
        s2riv1(k2)=sal*EXP(sal2r7(k2))
        rivsum=s2riv1(k2)
        sal2mean(k2)=salo-s2riv1(k2)
        a4402(k2)=dominter-domslope*sal2mean(k2)
        if(sal2mean(k2).GE.salmax)a4402(k2)=domback
        rriv2(ipart2,k2)=qs*EXP(revr72(ipart2,k2))
        pnr2(ipart2,k2)=(6.0/(pi*dip**3))*rriv2(ipart2,k2)/rhosr
        else
        pnr2(ipart2,k2)=0
        endif
C**
C**
        if(riv2.EQ.1)then
        s2riv2(k2)=sal2*EXP(sal2r7(k2))
        rivsum=s2riv1(k2)+s2riv2(k2)
        if(rivsum.GT.salo)then
        rivsum=salo
        endif
        sal2mean(k2)=salo-rivsum
        a4402(k2)=dominter-domslope*sal2mean(k2)
        if(sal2mean(k2).GE.salmax)a4402(k2)=domback
        rriv2r2(ipart2,k2)=qs2*EXP(revr72(ipart2,k2))
        pnr2r2(ipart2,k2)=(6.0/(pi*dip**3))*rriv2r2(ipart2,k2)/rhosr2
        else
```

```
        pnr2r2(ipart2,k2)=0
        endif
c
        if(bot.EQ.1)then
        rbot2(ipart2,k2)=rt*EXP(r72(ipart2,k2))
        pnb2(ipart2,k2)=(6.0/(pi*dip**3))*rbot2(ipart2,k2)/rhos
        else
        pnb2(ipart2,k2)=0
        endif
C**
C**
        pnt2(ipart2,k2)=pns2(ipart2,k2)+pnr2(ipart2,k2)+pnb2(ipart2,k2)
     &+pnr2r2(ipart2,k2)
        if(pnt2(ipart2,k2).GT.9999999)pnt2(ipart2,k2)=9999999
        if(pnt2(ipart2,k2).LT.-1000)pnt2(ipart2,k2)=-55555
c
2002    continue
c
2005    continue
c
c
4001    continue
4000    continue
c
c
c*****************************************************************
c
        fw=EXP(5.2*((2.5*dip/d0)**0.2)-6.0)
c
        delta=0.72*d0*(2.5*dip/d0)**0.25
        deltab=0.72*dxb*(2.5*dip/dxb)**0.25
        deltar=0.72*dxbave(iy)*(2.5*dip/dxbave(iy))**0.25
        deltas=0.72*dxbave(iy)*(0.000006625/dxbave(iy))**0.25
c
c
c......for the outfall sediment flux use running mean values of ub *dxb
      shieldr=ubave(iy)**2/((rhos/ro-1.0)*g*dip)
      epsilonr_l=0.00035*ak3_l*((ubave(iy)/w0)**0.68)*
     &(shieldr**0.4)*deltar*g*per
      if(epsilonr_l.LT.0.0000000001)epsilonr_l = 0.0000000001
      epsilonr=epsilonr_l+epsilonw
c
c......for the accretion wave use running mean values of ub *dxb
```

```
      shieldrs=ubave(iy)**2/(.03*g*dr)
      epsilonrs_l=0.00035*ak3s*((ubave(iy)/w0sal)**0.68)*
     &(shieldrs**0.4)*deltas*g*per
      if(epsilonrs_l.LT.0.0000000001)epsilonrs_l = 0.0000000001
      epsilonrs=epsilonrs_l+epsilonw
c
c
       shieldb=ub**2/((rhos/ro-1.0)*g*dip)
       epsilonb_l=0.00035*ak3*((ub/w0)**0.68)*
     &(shieldb**0.4)*deltab*g*per
       if(epsilonb_l.EQ.0)epsilonb_l=0.0000000001
      epsilonb=epsilonb_l+epsilonw
c
c.......offshore
       shield=u**2/((rhos/ro-1.0)*g*dip)
       epsilon_l=0.00035*ak2*((u/w0)**0.68)*
     &(shield**0.4)*delta*g*per
       if(epsilon_l.LT.0.0000000001)epsilon_l = 0.0000000001
      epsilon=epsilon_l+epsilonw
c
c
c.....at the local break point for salinity
       epsilonbs_l=0.00035*ak3s*((ub/w0sal)**0.68)
     &*(shieldrs**0.4)*deltas*g*per
       if(epsilonbs_l.LT.0000000001)epsilonbs_l = 0.0000000001
      epsilonbs=epsilonbs_l+epsilonw
c
c......outfall shields and epsilon calculated in RIVERM subroutine
c
c
c
c
       if(riv.EQ.1)then
       CALL RIVERM(epsilonr,dr,rwidth,q,ux1,uy1,ix,iy,irx,iry,sx,
     &sy,rrsfix,ak5,w0,delta,rhosr,ro,dip,g,per,ak3,time,qsold,
     &depthc,tconriv,qs)
c
       CALL SALM1(epsilonrs,dr,rwidth,q,ux1,uy1,ix,iy,irx,iry,sx,
     &sy,salo,ak5s,w0sal,deltas,rhosr,ro,dip,g,per,ak3,time,salold,
     &depthc,tconsal,sal)
c
       else
       continue
```

```
      endif
c
C
      if(riv2.EQ.1)then
      CALL RIVER2M(epsilonr,dr2,rwidth2,q2,ux1,uy1,ix,iy,irx2,iry2,sx,
     &sy,rrsfix2,ak5,w0,deltab,rhosr2,ro,dip,g,per,ak3,time,qs2old,
     &depthc,tconriv,qs2)
c
      CALL SALM2(epsilonrs,dr2,rwidth2,q2,ux1,uy1,ix,iy,irx2,iry2,sx,
     &sy,salo,ak5s,w0sal,deltas,rhosr2,ro,dip,g,per,ak3,time,salold2,
     &depthc,tconsal,sal2)
c
      else
      continue
      endif
c
c
      if(surf.EQ.1)then
      CALL SURFLM(epsilonb,bd1,ba1,bh1,ibpx,ix,iy,
     &ak4,aks,akb,dip,w0,pi,g,rhos,ro,tanbeta,rcspfix,time,rsold,
     &depthc,tcon,rs)
      else
      continue
      endif
c
c
      if(bot.EQ.1)then
      CALL BOTTOMM(ix,iy,u,shield,fw,ak,pi,wndep,rcfix,time,rtold,
     &depthc,tcon,rt,w0,epsilon)
      else
      continue
      endif
c
c
c
     if(verdat.EQ.1)datum=depthc
     if(verdat.NE.1)datum=delta
c
     if(verdat.EQ.1)datmod=-1.0
     if(verdat.NE.1)datmod=1.0
c
      do 200 k=1,4
      if(k.EQ.1)zw(k)=depthc-100.0
```

```
        if(k.EQ.2)zw(k)=delta+100.0
        if(k.EQ.3)zw(k)=depthc-alay3
        if(k.EQ.4)zw(k)=depthc-alay4
c
       if(zw(k).GT.depthc)zw(k)=depthc
       if(zw(k).LT.delta)zw(k)=delta
        dep(k)=depthc-zw(k)
c
c ....erosion calculations
        salmean(k)=salo
        a440(k)=dominter-domslope*salmean(k)
        if(salmean(k).GE.salmax)a440(k)=domback
         if(dmix.GE.dep(k))then
        salr7(k)=(-1.0)*dep(k)*aksal*w0sal/epsilonbs
        else
        salr7(k)=(-1.0)*dep(k)*aksal*((dimx/4.21)**2.4)
     &*w0sal/epsilonbs_l
        endif
c
         if((depthc-dmix).GE.zw(k))then
        r7(ipart,k)=(-1.0)*zw(k)*w0/epsilon_l
        else
        r7(ipart,k)=(-1.0)*zw(k)*w0*(dmix/dip)**2.4/epsilon
c**************my fix 10-21-94*************
        if(k.EQ.2)r7(ipart,k)=(-1.0)*zw(k)*w0/epsilon_l
c********************
        endif
         if((depthc-dmix).LT.0.0)r7(ipart,k)=(-1.0)
     &*zw(k)*w0/epsilon
c
         if(dmix.GE.dep(k))then
        revr7(ipart,k)=(-1.0)*dep(k)*w0/epsilonb
        else
        revr7(ipart,k)=(-1.0)*dep(k)*w0*
     &(dmix/dip)**2.4/epsilonb_l
      endif
c**
c
        if(surf.EQ.1)then
        rsurf(ipart,k)=rs*EXP(r7(ipart,k))
        pns(ipart,k)=(6.0/(pi*dip**3))*rsurf(ipart,k)/rhos
        else
        pns(ipart,k)=0
```

```
      endif
C
      if(riv.EQ.1)then
      sriv1(k)=sal*EXP(salr7(k))
      rivsum=sriv1(k)
      salmean(k)=salo-sriv1(k)
      a440(k)=dominter-domslope*salmean(k)
      if(salmean(k).GE.salmax)a440(k)=domback
      rriv(ipart,k)=qs*EXP(revr7(ipart,k))
      pnr(ipart,k)=(6.0/(pi*dip**3))*rriv(ipart,k)/rhosr
      else
      pnr(ipart,k)=0
      endif
C
C
      if(riv2.EQ.1)then
      sriv2(k)=sal2*EXP(salr7(k))
      rivsum=sriv1(k)+sriv2(k)
      if(rivsum.GT.salo)then
      rivsum=salo
      endif
      salmean(k)=salo-rivsum
      a440(k)=dominter-domslope*salmean(k)
      if(salmean(k).GE.salmax)a440(k)=domback
      rrivr2(ipart,k)=qs2*EXP(revr7(ipart,k))
      pnrr2(ipart,k)=(6.0/(pi*dip**3))*rrivr2(ipart,k)/rhosr2
      else
      pnrr2(ipart,k)=0
      endif
c
C
      if(bot.EQ.1)then
      rbot(ipart,k)=rt*EXP(r7(ipart,k))
      pnb(ipart,k)=(6.0/(pi*dip**3))*rbot(ipart,k)/rhos
      else
      pnb(ipart,k)=0
      endif
C
      pnt(ipart,k)=pns(ipart,k)+pnr(ipart,k)+pnb(ipart,k)
     &+pnrr2(ipart,k)
c
200   CONTINUE
c
```

```
1095    FORMAT(F8.5,3x,F8.3,3x,F17.4,3x,F12.8,f12.8)
c
499     continue
c
c.........sum numbers from each size bin together
        suml1(ix,iy)=pnt(ipart,1)+suml1(ix,iy)
        suml2(ix,iy)=pnt(ipart,2)+suml2(ix,iy)
        suml3(ix,iy)=pnt(ipart,3)+suml3(ix,iy)
        suml4(ix,iy)=pnt(ipart,4)+suml4(ix,iy)
c
c
        write(2, REC=icount)rt,rs,qs,qs2,sal,sal2
1776    continue
c
c
c*******************null point write********************************
c..........null point write commented out
c.....       if(iy.NE.igrdy)go to 6005
c..      write(50,3000)rc(1),rc(2),rc(3),rc(4),rc(5),rc(6),rc(7),
c..    &rc(8),rc(9),unp
c..3000    format(10e10.3)
c..6005    continue
c*******************************************************************
c
c
c.....put salinity values int x-y array
        sall1(ix,iy)=salmean(1)
        sall2(ix,iy)=salmean(2)
        sall3(ix,iy)=salmean(3)
        sall4(ix,iy)=salmean(4)
        acdom1(ix,iy)=a440(1)
        acdom2(ix,iy)=a440(2)
        acdom3(ix,iy)=a440(3)
        acdom4(ix,iy)=a440(4)
c
c
        if(suml1(ix,iy).GT.9999999)suml1(ix,iy)=9999999
        if(suml2(ix,iy).GT.9999999)suml2(ix,iy)=9999999
        if(suml3(ix,iy).GT.9999999)suml3(ix,iy)=9999999
        if(suml4(ix,iy).GT.9999999)suml4(ix,iy)=9999999
c
c
        if(suml1(ix,iy).LT.-1000)suml1(ix,iy)=-1000
```

```
        if(suml2(ix,iy).LT.-1000)suml2(ix,iy)=-1000
        if(suml3(ix,iy).LT.-1000)suml3(ix,iy)=-1000
        if(suml4(ix,iy).LT.-1000)suml4(ix,iy)=-1000
c
c
c...add fractional grains to the first 5 bins at surface & bottom
        pnt(1,1)=pnt(1,1)+an0b1
        pnt(2,1)=pnt(2,1)+an0b2
        pnt(3,1)=pnt(3,1)+an0b3
        pnt(4,1)=pnt(4,1)+an0b4
        pnt(5,1)=pnt(5,1)+an0b5
        pnt(6,1)=pnt(6,1)+an0b6
        pnt(7,1)=pnt(7,1)+an0b7
        pnt(8,1)=pnt(8,1)+an0b8
        pnt(9,1)=pnt(9,1)+an0b9
c
c
        pnt(1,2)=pnt(1,2)+an0b1
        pnt(2,2)=pnt(2,2)+an0b2
        pnt(3,2)=pnt(3,2)+an0b3
        pnt(4,2)=pnt(4,2)+an0b4
        pnt(5,2)=pnt(5,2)+an0b5
        pnt(6,2)=pnt(6,2)+an0b6
        pnt(7,2)=pnt(7,2)+an0b7
        pnt(8,2)=pnt(8,2)+an0b8
        pnt(9,2)=pnt(9,2)+an0b9
c
c
c
        kounts=0
       sumsp=0
       do 1790 is=1,3
       isp1=is+1
       sp(is)=ABS((LOG(pnt(isp1,1)/pnt(is,1)))/(LOG(d(isp1)/d(is))))
       sumsp=sumsp+sp(is)
       if(sp(is).LT.0.1)go to 1789
       kounts=kounts+1
1789   continue
1790   continue
c
       kountb=0
       sumsp2=0
       do 1788 is=1,3
```

```
      isp1=is+1
      sp2(is)=ABS((LOG(pnt(isp1,2)/pnt(is,2)))/(LOG(d(isp1)/d(is))))
      sumsp2=sumsp2+sp2(is)
      if(sp2(is).LT.0.1)go to 1787
      kountb=kountb+1
1787  continue
1788  continue
c
      b(ix,iy)=sumsp/(kounts+1)
      b2(ix,iy)=sumsp2/(kountb+1)
      if(b(ix,iy).GT.gamma)b(ix,iy)=gamma
      if(b(ix,iy).LT.1.0)b(ix,iy)=1.0
      if(b2(ix,iy).GT.gamma)b2(ix,iy)=gamma
      if(b2(ix,iy).LT.1.0)b2(ix,iy)=1.0
c
1791   continue
c
c
c****************selected grid loop start with ix,iy EQ igrdx igrdy***
      if(iy.NE.igrdy)go to 4002
      if(ix.NE.igrdx)go to 4003
        do 2006 k3=1,ires

c.....initialize sum2(k3) to background level n0
       sum2(k3)=n0
        do 2007 ipart3=1,ibins
c
c-----------------Slope Calculation for selected grid point------------
c...add fractional grains to the first 5 bins at each point in water
       pnt2(1,k3)=pnt2(1,k3)+an0b1
       pnt2(2,k3)=pnt2(2,k3)+an0b2
       pnt2(3,k3)=pnt2(3,k3)+an0b3
       pnt2(4,k3)=pnt2(4,k3)+an0b4
       pnt2(5,k3)=pnt2(5,k3)+an0b5
       pnt2(6,k3)=pnt2(6,k3)+an0b6
       pnt2(7,k3)=pnt2(7,k3)+an0b7
       pnt2(8,k3)=pnt2(8,k3)+an0b8
       pnt2(9,k3)=pnt2(9,k3)+an0b9
c
c
c.........sum numbers from each size bin together for selected location
       sum2(k3)=pnt2(ipart3,k3)+sum2(k3)
       if(sum2(k3).GT.9999999)sum2(k3)=9999999
```

```
2007     continue
c
c
2006    continue
c
c
       do 1462 k4=ires,1,-1
c.............calculate slope of profile..............
       sumspp=0
       kountp=0
c
       do 1795 is=1,3
       isp1=is+1
       spp(is)=ABS((LOG(pnt2(isp1,k4)/
     &pnt2(is,k4)))/(LOG(d(isp1)/d(is))))
       sumspp=sumspp+spp(is)
       if(spp(is).LT.0.1)go to 1794
       kountp=kountp+1
1794   continue
1795   continue
c
       slope(k4)=sumspp/(kountp+1)
       if(slope(k4).GT.gamma)slope(k4)=gamma
       if(slope(k4).LT.1.0)slope(k4)=1.0
       write(99,3001)dep2(k4),char(9),sum2(k4),char(9),
     &slope(k4),char(9),sal2mean(k4),char(9),a4402(k4)
1462    continue
4003     continue
4002     continue
c*******************end write selected grid pt file*****************
c
1801    continue
479     continue
c
        ixdum=100
       write(*,*)ixdum,iy,suml1(100,iy)
       write(*,*)epsilonb
c
1800    continue
c
1096    format(200f8.4)
1097    format(200f10.0)
1098    format(200f8.3)
```

```
3001    format(f9.3,1a,f10.0,1a,f8.3,1a,f8.3,1a,f8.4)
c
        do 1935 iy=ny,1,-1
        write(81,1098) (sall1(ix,iy),ix=1,nx)
        write(85,1096) (acdom1(ix,iy),ix=1,nx)
        write(82,1098) (sall2(ix,iy),ix=1,nx)
        write(86,1096) (acdom2(ix,iy),ix=1,nx)

        if(two_layer.NE.0)then
        write(83,1098) (sall3(ix,iy),ix=1,nx)
        write(87,1096) (acdom3(ix,iy),ix=1,nx)
        write(84,1098) (sall4(ix,iy),ix=1,nx)
        write(88,1096) (acdom4(ix,iy),ix=1,nx)
        endif
1935    continue
c
        do 1802 ix=1,nx
        write(41,1097) (suml1(ix,iy),iy=1,ny)
        write(91,1098) (b(ix,iy),iy=1,ny)
        write(42,1097) (suml2(ix,iy),iy=1,ny)
        write(92,1098) (b2(ix,iy),iy=1,ny)
c
        if(two_layer.NE.0)then
        write(43,1097) (suml3(ix,iy),iy=1,ny)
        write(44,1097) (suml4(ix,iy),iy=1,ny)
        endif
c
1802    continue
c
c
        stop
        END
c
c**********************Outfall Subroutine********************************
      SUBROUTINE RIVERM(epsilonr,dr,rwidth,q,ux1,uy1,ix,iy,irx,iry,sx,
     &sy,rrsfix,ak5,w0,deltab,rhosr,ro,dip,g,per,ak3,time,qsold,
     &depthc,tconriv,qs)
c
c
        drc=dr*100.0
        rwidthc=rwidth*100.0
        qc=q*1000000.0
c
```

```
c...modification to produce sediment plots using variable current
c...................... (RADIANS)..............................
c
        uvdir=ATAN2(uy1,ux1)
        rxyr=((iry-iy+.000001)*sy)/(ABS(irx-ix+.000001)*sx)
       thta=uvdir+ATAN(rxyr)
       ut=qc/(rwidthc*drc)
      ur=ut*COS(thta)-((ux1**2+uy1**2)**0.5)*SIN(thta)
c.................................................................
c
c
       e2r=ak5*(ur-(ur**2+4*w0*epsilonr)**0.5)/(2*epsilonr)
c
       rsr1=(ABS(ut*rrsfix))/(ut**2+4*w0*epsilonr)**0.5
       rsr3=e2r*((((irx-ix)*sx)**2)+(((iry-iy)*sy)**2))**0.5
       rsr2=EXP(rsr3)
       timedec=(time*tconriv*w0/depthc)
       if(timedec.GT.1.0)timedec=1.0
       timedec2=0.03116*(ux1**2+uy1**2)**0.5
       qsnew=rsr1*rsr2
       qfosl=qsold*(1-timedec)
       if(qsnew.GT.qfosl)then
       qs=qsnew+qfosl
       else
       qs=qsnew+qfosl*EXP(-1.0*timedec2)
       endif
       return
       end
c
c
c***********************End Outfall Subroutine*******************
c
c
       SUBROUTINE SURFLM(epsilonb,bd1,ba1,bh1,ibpx,ix,iy,
     &ak4,aks,akb,dip,w0,pi,g,rhos,ro,tanbeta,rcspfix,time,
     &rsold,depthc,tcon,rs)
c
c*********************Begin Surfl Subroutine******************
c
       us=ABS((g*bd1)**0.5*COS(ba1)**2)
       e2=(us-(us**2+4*ak4*w0*epsilonb)**0.5)/(2*epsilonb)
       denom=pi*dip**3*(rhos-ro)*akb*bd1**2*9240.0/tanbeta
       rn=(24*aks*ro*bh1**2*(g*bd1)**0.5)/denom
```

```
      rr=pi*dip**3*rn*rhos/6
      rs1=(rr/(us**2+4*w0*epsilonb)**0.5)
      rs3=(e2*ABS(ibpx-ix)*77.5)
      rs2=EXP(rs3)
      rs=rs1*rs2*rcspfix+rsold*(1-(EXP(-1.0*depthc*tcon/(2*time*w0))))
c
      return
      end
c
c***************End Surfl Subroutine****************************
c
c
      SUBROUTINE BOTTOMM(ix,iy,u,shield,fw,ak,pi,wndep,rcfix,time,
     &rtold,depthc,tcon,rt,w0,epsilon)
c
*************************** Begin Bottom Subroutine***************
c
       if (u.LT..000001) then
       nomo=1
       rt1=0.0
       go to 399
       endif
c
      rt5=shield*fw
      if(rt5.le..05)then
      nomo=1
      rt1=0.0
      go to 399
      endif
c
c
      rt2=(0.05/(rt5))**.5
      rt4=ACOS(rt2)
      rt1=ak*(fw*shield-0.05)*(2/pi)*rt4
399   continue
      rt=rt1+(rtold*(1-(EXP(-1.0*depthc*tcon/(2*time*w0)))))
c
c*************rcfix section was commented out Sept 12, 1994 10:00
      if(rt.GT.rcfix)then
      rt1=rcfix
      rt=rt1+(rtold*(1-(EXP(-1.0*depthc*tcon/(2*time*w0)))))
      endif
c
```

```
c
      return
      end
c
c*********************End Bottom Subroutine************************
c
c***********************Outfall2 Subroutine********************************
      SUBROUTINE RIVER2M(epsilonr,dr2,rwidth2,q2,ux1,uy1,ix,iy,irx2,iry2
     &,sx,sy,rrsfix2,ak5,w0,deltab,rhosr2,ro,dip,g,per,ak3,time,
     &qs2old,depthc,tconriv,qs2)
c
c
       drc=dr2*100.0
       rwidthc=rwidth2*100.0
       qc=q2*1000000.0
c
c...modification to produce sediment plots using variable current
c...................... (RADIANS)..............................
c
       uvdir=ATAN2(uy1,ux1)
       rxyr=((iry2-iy+.000001)*sy)/(ABS(irx2-ix+.000001)*sx)
      thta=uvdir+ATAN(rxyr)
      ut=qc/(rwidthc*drc)
     ur=ut*COS(thta)-((ux1**2+uy1**2)**0.5)*SIN(thta)
c...................................................................
c
      e2r=ak5*(ur-(ur**2+4*w0*epsilonr)**0.5)/(2*epsilonr)
c
      rsr1=(ABS(ut*rrsfix2))/(ut**2+4*w0*epsilonr)**0.5
      rsr3=e2r*((((irx2-ix)*sx)**2)+(((iry2-iy)*sy)**2))**0.5
      rsr2=EXP(rsr3)
      timedec=(time*tconriv*w0/depthc)
      if(timedec.GT.1.0)timedec=1.0
      timedec2=0.03116*(ux1**2+uy1**2)**0.5
      qsnew=rsr1*rsr2
      qfosl=qs2old*(1-timedec)
      if(qsnew.GT.qfosl)then
      qs2=qsnew+qfosl
      else
      qs2=qsnew+qfosl*EXP(-1.0*timedec2)
      endif
c
      return
```

```
      end
c
c
c************************End Outfall2 Subroutine********************
c
c*****************Accretion Outfall 1 Subroutine************************
      SUBROUTINE SALM1(epsilonrs,dr,rwidth,q,ux1,uy1,ix,iy,irx,iry,sx,
     &sy,salo,ak5s,w0sal,deltas,rhosr,ro,dip,g,per,ak3,time,salold,
     &depthc,tconsal,sal)
c
c
       drc=dr*100.0
       rwidthc=rwidth*100.0
       qc=q*1000000.0
c
c...modification to produce sediment plots using variable current
c...................... (RADIANS)..............................
c
       uvdir=ATAN2(uy1,ux1)
       rxyr=((iry-iy+.000001)*sy)/(ABS(irx-ix+.000001)*sx)
      thta=uvdir+ATAN(rxyr)
      ut=qc/(rwidthc*drc)
     ur=ut*COS(thta)-((ux1**2+uy1**2)**0.5)*SIN(thta)
c...................................................................
c
c
      e2r=ak5s*(ur-(ur**2+4*w0sal*epsilonrs)**0.5)/(2*epsilonrs)
c
      rsr1=(ABS(ut*salo))/(ut**2+4*w0sal*epsilonrs)**0.5
      rsr3=e2r*((((irx-ix)*sx)**2)+(((iry-iy)*sy)**2))**0.5
      rsr2=EXP(rsr3)
      timedec=(time*tconsal*w0sal/depthc)
      if(timedec.GT.1.0)timedec=1.0
      timedec2=0.03116*(ux1**2+uy1**2)**0.5
      snew=rsr1*rsr2
      sfosl=salold*(1-timedec)
      if(snew.GT.sfosl)then
      sal=snew+sfosl
      else
      sal=snew+sfosl*EXP(-1.0*timedec2)
      endif
      return
      end
```

```
c
c
c*************End Accretion Outfall 1 Subroutine*******************
c
c
c***************Salinity Outfall2 Subroutine************************
      SUBROUTINE SALM2(epsilonrs,dr2,rwidth2,q2,ux1,uy1,ix,iy,irx2,iry2
     &,sx,sy,salo,ak5s,w0sal,deltas,rhosr2,ro,dip,g,per,ak3,time,
     &salold2,depthc,tconsal,sal2)
c
c
        drc=dr2*100.0
        rwidthc=rwidth2*100.0
        qc=q2*1000000.0
c
c...modification to produce sediment plots using variable current
c....................... (RADIANS)...............................
c
        uvdir=ATAN2(uy1,ux1)
        rxyr=((iry2-iy+.000001)*sy)/(ABS(irx2-ix+.000001)*sx)
       thta=uvdir+ATAN(rxyr)
       ut=qc/(rwidthc*drc)
      ur=ut*COS(thta)-((ux1**2+uy1**2)**0.5)*SIN(thta)
c....................................................................
c
c
       e2r=ak5s*(ur-(ur**2+4*w0sal*epsilonrs)**0.5)/(2*epsilonrs)
c
       rsr1=(ABS(ut*salo))/(ut**2+4*w0sal*epsilonrs)**0.5
       rsr3=e2r*((((irx2-ix)*sx)**2)+(((iry2-iy)*sy)**2))**0.5
       rsr2=EXP(rsr3)
       timedec=(time*tconsal*w0sal/depthc)
       if(timedec.GT.1.0)timedec=1.0
       timedec2=0.03116*(ux1**2+uy1**2)**0.5
       snew=rsr1*rsr2
       sfosl=salold2*(1-timedec)
       if(snew.GT.sfosl)then
       sal2=snew+sfosl
       else
       sal2=snew+sfosl*EXP(-1.0*timedec2)
       endif
c
       return
```

```
      end
c
c
c************************End Accretion Outfall2 Subroutine*************
```

APPENDIX F

CODE FOR THE MULTINODE DILUTION MODEL

Appendix F:
Code for the Multinode Dilution Model

```
c*********************************************************************
c
c      multinode.for - 18 June 2004
c**with verticle loop
c reads in raster 200 row 200 column arrays of:
c depths, wave heights, wave numbers, x- and y-directed current components
c Surface and Bottom Salinity & caluculates Dilution from these
c              written by Scott A. Jenkins & Joseph Wasyl
c
       character*26 infile1,infile2,infile4,infile5
c
       DIMENSION depth(200,200),wht(200,200),wvn(200,200)
       DIMENSION xcur(200,200),ycur(200,200),col(200,200)
c
      dimension z(5),s(5),ps1(5),ps1b(5),ps1bav(5)
      integer ipt(5),ix(5),iy(5)
      dimension pnx(200,200),pny(200,200),pn(200,200),pn_l(200,200)
      dimension brx(200,200),bry(200,200),brn(200,200),dmix(200,200)
      dimension salsur(200,200),salbrn(200,200),saldep(200,200)
      dimension dilriv(200,200),dildep(200,200),colriv(200,200)
      dimension dilbrn(200,200),dbrnav(200,200),sbrnav(200,200)
      dimension brxav(200,200),bryav(200,200),brnav(200,200)
      character*12 ifile,ofile1,ofile2,ofile3,ofile4,ofile5,ofile6
      character*12 ofile7,ofile8,ofile9,ofil10,ofil11,ofil12,ofil13
c
c
     open(42,file='multinode_agua_3.inp',status='old')
       read(42,3000)ifile
       read(42,3000)infile1
       read(42,3111)infile2
       read(42,3200)infile4
       read(42,3200)infile5
       read(42,3000)ofile1
       read(42,3000)ofile2
       read(42,3000)ofile3
       read(42,3000)ofile4
       read(42,3000)ofile5
       read(42,3000)ofile6
       read(42,3000)ofile7
       read(42,3000)ofile8
```

```
read(42,3000)ofile9
read(42,3000)ofil10
read(42,3000)ofil11
read(42,3000)ofil12
read(42,3000)ofil13
read(42,*)num
read(42,*)imax
read(42,*)jmax
read(42,*)srivx1
read(42,*)srivx2
read(42,*)srivx3
read(42,*)srivx4
read(42,*)srivx5
read(42,*)srivy1
read(42,*)srivy2
read(42,*)srivy3
read(42,*)srivy4
read(42,*)srivy5
read(42,*)ub
read(42,*)vb
read(42,*)u2
read(42,*)v2
read(42,*)sx
read(42,*)sy
read(42,*)akx1_1
read(42,*)aky1_1
read(42,*)akx1_2
read(42,*)aky1_2
read(42,*)akx1_3
read(42,*)aky1_3
read(42,*)akx1_4
read(42,*)aky1_4
read(42,*)akx1_5
read(42,*)aky1_5
read(42,*)ak2_1
read(42,*)ak2_2
read(42,*)ak2_3
read(42,*)ak2_4
read(42,*)ak2_5
read(42,*)akz_1
read(42,*)akz_2
read(42,*)akz_3
read(42,*)akz_4
```

```
read(42,*)akz_5
read(42,*)thresh
read(42,*)back
read(42,*)salo
read(42,*)cline
read(42,*)isal
read(42,*)pnman
read(42,*)upper
read(42,*)brine
read(42,*)upperb
read(42,*)ibrn
read(42,*)brnman
read(42,*)per
read(42,*)ruf
read(42,*)rhos
read(42,*)wind
read(42,*)deglat
read(42,*)ibadx
read(42,*)ibady
read(42,*)qsa
read(42,*)qtc
read(42,*)widsa
read(42,*)widtc
read(42,*)akr_3
read(42,*)akr_4
read(42,*)u_blend
read(42,*)aksal
read(42,*)surdz
read(42,*)botdz
read(42,*)shrdec
read(42,*)swrdec
read(42,*)surmix
read(42,*)aksur
read(42,*)surhb
read(42,*)thermo
read(42,*)xmag
read(42,*)ymag
read(42,*)akocs
read(42,*)depo
read(42,*)thrmdz
read(42,*)coli
read(42,*)deep
read(42,*)akzav
```

```
c
       rawbrn=2.0*salo
C**read in oceanbat .grd, oceanrds_200x200_raster and tidecur_agua arrays
c
      OPEN(UNIT=51,FILE=infile1,STATUS='OLD')
      OPEN(UNIT=52,FILE=infile2,STATUS='OLD')
      OPEN(UNIT=54,FILE=infile4,STATUS='OLD')
      OPEN(UNIT=55,FILE=infile5,STATUS='OLD')
C
      DO 8001 J=1,200
      READ(51,*)(depth(I,J),I=1,200)
      READ(52,*)(wht(I,J),I=1,200)
      READ(54,*)(xcur(I,J),I=1,200)
      READ(55,*)(ycur(I,J),I=1,200)
8001  CONTINUE
c
3000    format(a12)
3111    format(a26)
3200    format(a22)
c
       surhb=surhb*100.0
       surdz=surdz*100.0
       botdz=botdz*100.0
       cline=cline*100.0
       depo=depo*100.0
       thrmdz=thrmdz*100.0
       freq=1.0/per
       pi=ACOS(-1.0)
       g=980.0
       rho=1.03
       sigma=2.0*pi*freq
       deepl=2.0*pi*g/(sigma**2.0)
       wind=wind*0.5148*100.0
       ak8=0.00009/((wind**0.5)+1.1103)
       ustar=(ak8*(wind**2))**0.5
       rot_e=0.000072685
       f=2.0*rot_e*SIND(deglat)
       dmix_test=4.0*ustar/(f*100.0)
c.......epsilonw is in cgs units
       epsilonw=0.0000043*(wind**2.0)
       if(wind.LE.600)epsilonw=0.0000000102*(wind**3.0)
c
c
```

```
      open(20,file=ifile,status='old')
      open(21,file=ofile1,status='unknown')
      open(22,file=ofile2,status='unknown')
      open(23,file=ofile3,status='unknown')
      open(24,file=ofile4,status='unknown')
      open(25,file=ofile5,status='unknown')
      open(26,file=ofile6,status='unknown')
      open(27,file=ofile7,status='unknown')
      open(28,file=ofile8,status='unknown')
      open(29,file=ofile9,status='unknown')
      open(30,file=ofil10,status='unknown')
      open(31,file=ofil11,status='unknown')
      open(32,file=ofil12,status='unknown')
      open(33,file=ofil13,status='unknown')
c
c
c... open node file
       do 100 i=1,num
       read(20,1000) ipt(i),ix(i),iy(i),z(i)
100    continue
1000    format(i2,2i4,f7.2)
c
c..covert x and y coordinates to nearest integer
c..calculate or assign strength to each source
c
c**initialize particle output array
       do 870 iay=1,jmax
       do 860 iax=1,imax
       pn(iax,iay)=0.0
       pnx(iax,iay)=0.0
       pny(iax,iay)=0.0
       brn(iax,iay)=0.0
       brx(iax,iay)=0.0
       bry(iax,iay)=0.0
       brnav(iax,iay)=0.0
       brxav(iax,iay)=0.0
       bryav(iax,iay)=0.0
       xcur(iax,iay)=xmag*xcur(iax,iay)
       ycur(iax,iay)=ymag*ycur(iax,iay)
c
       depth(iax,iay)=depth(iax,iay)*100.0
c**make land in bays negative in increasing iax direction
       if(iland.EQ.1)depth(iax,iay)=-200.0
```

```
        if(depth(iax,iay).LE.0.0)then
        iland=1
        depth(iax,iay)=-200.0
        endif
c
c**dispersion routine for wave number
c.........initialize wave number array
         wvn(iax,iay)=-1.0
         amp2=0.05*deepl
         if(depth(iax,iay).LE.0)go to 1090
         b=(sigma**2*depth(iax,iay))/g
         if(depth(iax,iay).LT.amp2)then
          wvn(iax,iay)=sigma/(g*depth(iax,iay))**0.5
          go to 1090
          else
          endif
          a=b
         do 1030 k5=1,100
          h=tanh(a)
         fjw=b-a*h
         if (abs(fjw) .lt. 0.000001) go to 1040
         fd=-1.0*h-(a/cosh(a)**2)
         a=a-(fjw/fd)
1030     continue
         write(*,1050)
1050     format(' subroutine disp does not converge!!! ')
1040      wvn(iax,iay)=a/depth(iax,iay)
1090     continue
cEND DISPERSION SUBROUTINE
        wht(iax,iay)=wht(iax,iay)*100.0
c
860     continue
        iland=0
870     continue
c
c&&&&&&&&&&
        do 200 i=1,num
        ps1b(i)=0.0
        ps1bav(i)=0.0
200     continue
c
cPick each Source, use x and y location arrays ix(i), iy(i) of source
        do 799 i=1,num
```

```
cPick each x location of grid(iax) solve dx relative to x location of source ix(i)
      do 800 iax=1,imax
      dx=(iax-ix(i))*sx
      do 850 iay=1,jmax
      dmix(iax,iay)=4.0*ustar/(f*100.0)
      if(dmix(iax,iay).LE.cline)dmix(iax,iay)=cline
      if(depth(iax,iay).GT.0.0.AND.xcur(iax,iay).EQ.0.0)
     &xcur(iax,iay)=0.00001
      if(depth(iax,iay).GT.0.0.AND.ycur(iax,iay).EQ.0.0)
     &ycur(iax,iay)=0.00001
      dy=(iay-iy(i))*sy
      dr=((dx**2.0)+(dy**2.0))**0.5
      drxy=((sx**2.0)+(sy**2.0))**0.5
      if(dr.LT.drxy)dr=drxy
      theta=ABS(dx/dr)
      d0=wht(iax,iay)/SINH(wvn(iax,iay)*depth(iax,iay))+0.001
      uwav=sigma*d0
      ucur=(xcur(iax,iay)**2.0+ycur(iax,iay)**2.0)**0.5
      u=ucur+uwav
C------
      fw=EXP(5.2*((2.5*ruf/d0)**0.2)-6.0)
      delta=0.72*d0*(2.5*ruf/d0)**0.25
      shield=(u**2/((rhos/rho-1.0)*g*ruf))*fw
C-----
C************SIO_1 OUTFALL X-LOOP (ipt=1) lat=33 39' 19"N, lon=117 58' 57"W*****
      if(ipt(i).EQ.1)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      ps1bav(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4521
      endif
      s(i)=srivx1
      ak1=akx1_1
      ak2=ak2_1
      surdec=-1.0*((depth(iax,iay)/surhb)**aksur)
      epsur=surmix*(EXP(surdec))
c
c*U_BLEND logic
      uave=(u_blend*ucur+u)/(u_blend+1)
      dxb=uave*per
```

```
      deltar=0.72*dxb*(2.5*ruf/dxb)**0.25
      fw=EXP(5.2*((2.5*ruf/dxb)**0.2)-6.0)
      shield=(uave**2/((rhos/rho-1.0)*g*ruf))*fw
      epsilon_l=0.00035*ak2*(uave**0.68)*(shield**0.4)*deltar*g*per
      if(xcur(iax,iay).LT.0.0)then
      uwav=-1.0*uave
      else
      uwav=uave
      endif
      u0=xcur(iax,iay)+uwav+ub
      u0abs=ABS(u0)
c
      if(epsilon_l.EQ.0.0)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      depmdz=depth(iax,iay)-botdz
      botx=-1.0*akz_1*(botdz**1.0)/eps
      botav=-1.0*akzav*(depth(iax,iay)**1.0)/eps
      decbot=EXP(botx)
      decav=EXP(botav)
      if(dmix(iax,iay).GE.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &dmix(iax,iay)=depth(iax,iay)
      if(dmix(iax,iay).GT.depmdz)decbot=decbot**shrdec
      ps1b(i)=s(i)*decbot/((uave**2+4.0*ruf*(eps+epsur))**0.5)
      ps1bav(i)=ps1b(i)*decav/decbot
c SIO_1 Inshore
      if(iax.EQ.237.AND.iay.EQ.133)write(*,*)iax,iay,epsilon_l,epsur,
     &ps1b(i),etrb
c SIO_1 Inshore
      if(iax.EQ.173.AND.iay.EQ.96)write(*,*)iax,iay,epsilon_l,epsur,
     &ps1b(i),etrb
c SIO_1 Inshore
      if(iax.EQ.179.AND.iay.EQ.94)write(*,*)iax,iay,epsilon_l,epsur,
     &ps1b(i),etrb
c SIO_1 Inshore
      if(iax.EQ.163.AND.iay.EQ.97)write(*,*)iax,iay,epsilon_l,epsur,
     &ps1b(i),etrb
      if(u0.GE.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*((u0-theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.GE.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*((u0+theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.LT.0.0)then
```

```
      etr=-1.0*ak1*(-1.0*(u0-theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      au0=ABS(u0)
      if(dx.EQ.0.0)then
      etr=-1.0*ak1*(au0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
c
      if(u0.GE.0.0.AND.dx.GT.0.0)then
      etrb=-1.0*ak1*((u0-theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.GE.0.0.AND.dx.LT.0.0)then
      etrb=-1.0*ak1*((u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.LT.0.0)then
      etrb=-1.0*ak1*(-1.0*(u0-theta*u0)+(4.0*ruf*eps)**0.5)/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.GT.0.0)then
      etrb=-1.0*ak1*(-1.0*(u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      au0=ABS(u0)
      if(dx.EQ.0.0)then
      etrb=-1.0*ak1*(au0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4521  continue
c*********************END SIO_1 OUTFALL X-
LOOP******************************
c
C*********SIO_3 OUTFALL X-LOOP (ipt=2) ***************************
      if(ipt(i).EQ.2)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
```

```
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4522
      endif
      if(xcur(iax,iay).LT.0.0)uwav=-1.0*uwav
      u0=xcur(iax,iay)+uwav+u2
      s(i)=srivx2
      ak1=akx1_2
      ak2=ak2_2
      epsilon_l=0.00035*ak2*(u**0.68)*(shield**0.4)*delta*g*per
      if(epsilon_l.EQ.0.0)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      dmxmdz=dmix(iax,iay)-thrmdz
      if(thrmdz.NE.0)then
      if(dmxmdz.GT.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &thrmdz=dmxmdz-depth(iax,iay)+thrmdz
      endif
      thrmx=-1.0*akz_2*(thrmdz**1.0)
      dcthrm=EXP(thrmx)
      if(depth(iax,iay).LE.depo)then
      ps1(i)=(s(i)*dcthrm/((u**2+4.0*ruf*eps)**0.5))*
     &((depth(iax,iay)/depo)**akocs)
      else
      ps1(i)=(s(i)*dcthrm/((u**2+4.0*ruf*eps)**0.5))*
     &((depth(iax,iay)/depo)**(akocs/deep))
      endif
      if(u0.GE.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*((u0-theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.GE.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*((u0+theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0-theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      au0=ABS(u0)
      if(dx.EQ.0.0)then
```

```
      etr=-1.0*ak1*(au0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4522  continue
c*********************END SIO_3 OUTFALL X-LOOP******************************
c
C*********SIO_2 Outfall X-LOOP (ipt=3) ***************************
      if(ipt(i).EQ.3)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4523
      endif
      drc=z(i)*100.0
      wid=widsa*100.0
      qc=qsa*92903.0
      uvdir=ATAN2(ycur(iax,iay),xcur(iax,iay))
      ut0=qc/(drc*wid)
      ut=ut0/(((((dx+0.00001)**2+(dy+0.00001)**2)**0.5)**akr_3)
      u0=xcur(iax,iay)-ut*COS(uvdir)
      uave=(u_blend*ucur+u)/(u_blend+1)
      dxb=uave*per
      deltar=0.72*dxb*(2.5*ruf/dxb)**0.25
      fw=EXP(5.2*((2.5*ruf/dxb)**0.2)-6.0)
      shield=(uave**2/((rhos/rho-1.0)*g*ruf))*fw
      s(i)=srivx3
      ak1=akx1_3
      ak2=ak2_3
      epsilon_l=0.00035*ak2*(uave**0.68)*(shield**0.4)*deltar*g*per
      if(epsilon_l.EQ.0.0)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
c***diagnostic
CCCCCCC      if(iay.EQ.118)write(*,*)iax,iay,ur,shield,deltar,eps
c***diagnostic
      surx=-1.0*akz_3*(surdz**1.0)
      decsur=EXP(surx)
      if(dmix(iax,iay).GE.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &dmix(iax,iay)=depth(iax,iay)
      if(dmix(iax,iay).LT.surdz)decsur=decsur**thermo
```

```
      ps1(i)=s(i)*decsur/((u0**2+4.0*ruf*eps)**0.5)
      if(u0.GE.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*((u0-theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.GE.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*((u0+theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0-theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      au0=ABS(u0)
      if(dx.EQ.0.0)then
      etr=-1.0*ak1*(au0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4523  continue
c*********************END  SIO_2 Outfall X-LOOP*****************************
c
C*********SIO_4  OUTFALL X-LOOP (ipt=4) ***************************
      if(ipt(i).EQ.4)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4524
      endif
      drc=z(i)*100.0
      wid=widtc*100.0
      qc=qtc*92903.0
      uvdir=ATAN2(ycur(iax,iay),xcur(iax,iay))
      ut0=qc/(drc*wid)
      ut=ut0/((((dx+0.00001)**2+(dy+0.00001)**2)**0.5)**akr_4)
      u0=xcur(iax,iay)-ut*COS(uvdir)
      uave=(u_blend*ucur+u)/(u_blend+1)
      dxb=uave*per
      deltar=0.72*dxb*(2.5*ruf/dxb)**0.25
```

```
      fw=EXP(5.2*((2.5*ruf/dxb)**0.2)-6.0)
      shield=(uave**2/((rhos/rho-1.0)*g*ruf))*fw
      s(i)=srivx4
      ak1=akx1_4
      ak2=ak2_4
      epsilon_l=0.00035*ak2*(uave**0.68)*(shield**0.4)*deltar*g*per
      if(epsilon_l.EQ.0.0000000001)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
c***diagnostic
CCCCCCC       if(iay.EQ.118)write(*,*)iax,iay,ur,shield,deltar,eps
c***diagnostic
      surx=-1.0*akz_4*(surdz**2.0)/eps
      decsur=EXP(surx)
      if(dmix(iax,iay).GE.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &dmix(iax,iay)=depth(iax,iay)
      if(dmix(iax,iay).LT.surdz)decsur=decsur**thermo
      ps1(i)=s(i)*decsur/((u0**2+4.0*ruf*eps)**0.5)
      if(u0.GE.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*((u0-theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.GE.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*((u0+theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0-theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      au0=ABS(u0)
      if(dx.EQ.0.0)then
      etr=-1.0*ak1*(au0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4524  continue
c*********************END SIO_4 X-LOOP*****************************
c
C*********SIO_ Offshore OUTFALL X-LOOP (ipt=5) ***************************
      if(ipt(i).EQ.5)then
      if(depth(iax,iay).LT.10.0)then
```

```
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4525
      endif
      if(xcur(iax,iay).LT.0.0)uwav=-1.0*uwav
      u0=xcur(iax,iay)+uwav
      s(i)=srivx5
      ak1=akx1_5
      ak2=ak2_5
      epsilon_l=0.00035*ak2*(u**0.68)*(shield**0.4)*delta*g*per
      if(epsilon_l.EQ.0.0000000001)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      ps1(i)=s(i)/((u**2+4.0*ruf*eps)**0.5)
      if(u0.GE.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*((u0-theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.GE.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*((u0+theta*u0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0-theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(u0.LT.0.0.AND.dx.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(u0+theta*u0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      au0=ABS(u0)
      if(dx.EQ.0.0)then
      etr=-1.0*ak1*(au0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4525  continue
c*********************END SIO Offshore OUTFALL X-LOOP******************************
c
      ps3=etr*((dx**2)+(dy**2))**0.5
      ps3b=etrb*((dx**2)+(dy**2))**0.5
c***
      ps2=EXP(ps3)
```

```
      ps2b=EXP(ps3b)
      pnx(iax,iay)=pnx(iax,iay)+(ps1(i)*ps2)
      brx(iax,iay)=brx(iax,iay)+(ps1b(i)*ps2b)
      brxav(iax,iay)=brxav(iax,iay)+(ps1bav(i)*ps2b)
c
c
850   continue
800   continue
799   continue
c
C**********V LOOPS **********
c
cPick each Source, use x and y location arrays ix(i), iy(i) of source
      do 1799 i=1,num
cPick each x location of grid(iax) solve dx relative to x location of source ix(i)
      do 1800 iay=1,jmax
      dy=(iay-iy(i))*sy
c
      do 1850 iax=1,imax
      dmix(iax,iay)=4.0*ustar/(f*100.0)
      if(dmix(iax,iay).LE.cline)dmix(iax,iay)=cline
      if(iax.EQ.ibadx.AND.iay.EQ.ibady)then
      ps1(i)=0.0
      ps1b(i)=0.0
      ps1bav(i)=0.0
      etr=0.0
      etrb=0.0
      go to 1777
      endif
      if(depth(iax,iay).GT.0.0.AND.xcur(iax,iay).EQ.0.0)
     &xcur(iax,iay)=0.00001
      if(depth(iax,iay).GT.0.0.AND.ycur(iax,iay).EQ.0.0)
     &ycur(iax,iay)=0.00001
      dx=(iax-ix(i))*sx
      dr=((dx**2.0)+(dy**2.0))**0.5
      drxy=((sx**2.0)+(sy**2.0))**0.5
      if(dr.LT.drxy)dr=drxy
      beta=ABS(dy/dr)
c
      d0=wht(iax,iay)/SINH(wvn(iax,iay)*depth(iax,iay))+0.001
      uwav=sigma*d0
      ucur=(xcur(iax,iay)**2.0+ycur(iax,iay)**2.0)**0.5
      u=ucur+uwav
```

```
C------
      fw=EXP(5.2*((2.5*ruf/d0)**0.2)-6.0)
      delta=0.72*d0*(2.5*ruf/d0)**0.25
      shield=(u**2/((rhos/rho-1.0)*g*ruf))*fw
C-----
c*************SIO_1 OUTFALL Y-LOOP***********************************
      if(ipt(i).EQ.1)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      ps1bav(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4530
      endif
      s(i)=srivy1
      ak1=aky1_1
      ak2=ak2_1
      surdec=-1.0*((depth(iax,iay)/surhb)**aksur)
      epsur=surmix*(EXP(surdec))
c
c* U_BLEND logic
      uave=(u_blend*ucur+u)/(u_blend+1)
      dxb=uave*per
      deltar=0.72*dxb*(2.5*ruf/dxb)**0.25
      fw=EXP(5.2*((2.5*ruf/dxb)**0.2)-6.0)
      shield=(uave**2/((rhos/rho-1.0)*g*ruf))*fw
      epsilon_l=0.00035*ak2*(uave**0.68)*(shield**0.4)*deltar*g*per
      if(ycur(iax,iay).LT.0.0)then
      uwav=-1.0*uave
      else
      uwav=uave
      endif
      v0=ycur(iax,iay)+uwav+vb
      v0abs=ABS(v0)
c
      if(epsilon_l.EQ.0.0)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      depmdz=depth(iax,iay)-botdz
      botx=-1.0*akz_1*(botdz**1.0)/eps
      botav=-1.0*akzav*(depth(iax,iay)**1.0)/eps
      decbot=EXP(botx)
      decav=EXP(botav)
```

```
      if(dmix(iax,iay).GE.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &dmix(iax,iay)=depth(iax,iay)
      if(dmix(iax,iay).GT.depmdz)decbot=decbot**shrdec
      ps1b(i)=s(i)*decbot/((uave**2+4.0*ruf*(eps+epsur))**0.5)
      ps1bav(i)=ps1b(i)*decav/decbot
      if(v0.GE.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*((v0-beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.GE.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*((v0+beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0-beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0+beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      av0=ABS(v0)
      if(dy.EQ.0.0)then
      etr=-1.0*ak1*(av0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
c
      if(v0.GE.0.0.AND.dy.GT.0.0)then
      etrb=-1.0*ak1*((v0-beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.GE.0.0.AND.dy.LT.0.0)then
      etrb=-1.0*ak1*((v0+beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.LT.0.0)then
      etrb=-1.0*ak1*(-1.0*(v0-beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.GT.0.0)then
      etrb=-1.0*ak1*(-1.0*(v0+beta*v0)+(4.0*ruf*eps)**0.5)/
     &(2.0*eps)
      endif
      av0=ABS(v0)
      if(dy.EQ.0.0)then
      etrb=-1.0*ak1*(av0+(4.0*ruf*eps)**0.5)/
     &(2.0*eps)
```

```
      endif
      endif
4530  continue
C*************************END SIO_1 Y-LOOP*********************************
c
c*************SIO_3 OUTFALL Y-LOOP**********************************
      if(ipt(i).EQ.2)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4531
      endif
      if(ycur(iax,iay).LT.0.0)uwav=-1.0*uwav
      v0=ycur(iax,iay)+uwav+v2
      s(i)=srivy2
      ak1=aky1_2
      ak2=ak2_2
      epsilon_l=0.00035*ak2*(u**0.68)*(shield**0.4)*delta*g*per
      if(epsilon_l.EQ.0.0000000001)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      dmxmdz=dmix(iax,iay)-thrmdz
      if(thrmdz.NE.0)then
      if(dmxmdz.GT.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &thrmdz=dmxmdz-depth(iax,iay)+thrmdz
      endif
      thrmx=-1.0*akz_2*(thrmdz**1.0)
      dcthrm=EXP(thrmx)
      if(depth(iax,iay).LE.depo)then
      ps1(i)=(s(i)/((u**2+4.0*ruf*eps)**0.5))*
     &((depth(iax,iay)/depo)**akocs)
      else
      ps1(i)=(s(i)/((u**2+4.0*ruf*eps)**0.5))*
     &((depth(iax,iay)/depo)**(akocs/deep))
      endif
      if(v0.GE.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*((v0-beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.GE.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*((v0+beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.LT.0.0)then
```

```
      etr=-1.0*ak1*(-1.0*(v0-beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0+beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      av0=ABS(v0)
      if(dy.EQ.0.0)then
      etr=-1.0*ak1*(av0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4531  continue
c
C***********************END SIO_3  OUTFALL Y-LOOP************************
c
c*************SIO_2 Outfall Y-LOOP**********************************
      if(ipt(i).EQ.3)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4532
      endif
      drc=z(i)*100.0
      wid=widsa*100.0
      qc=qsa*92903.0
      uvdir=ATAN2(ycur(iax,iay),xcur(iax,iay))
      ut0=qc/(drc*wid)
      ut=ut0/(((((dx+0.00001)**2+(dy+0.00001)**2)**0.5)**akr_3)
      v0=ycur(iax,iay)+ut*SIN(uvdir)
      uave=(u_blend*ucur+u)/(u_blend+1)
      dxb=uave*per
      deltar=0.72*dxb*(2.5*ruf/dxb)**0.25
      fw=EXP(5.2*((2.5*ruf/dxb)**0.2)-6.0)
      shield=(uave**2/((rhos/rho-1.0)*g*ruf))*fw
      s(i)=srivy3
      ak1=aky1_3
      ak2=ak2_3
      epsilon_l=0.00035*ak2*(uave**0.68)*(shield**0.4)*deltar*g*per
      if(epsilon_l.EQ.0.0)epsilon_l=0.0000000001
```

```
      eps=epsilon_l+epsilonw
      surx=-1.0*akz_3*(surdz**1.0)
      decsur=EXP(surx)
      if(dmix(iax,iay).GE.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &dmix(iax,iay)=depth(iax,iay)
      if(dmix(iax,iay).LT.surdz)decsur=decsur**thermo
      ps1(i)=s(i)*decsur/((v0**2+4.0*ruf*eps)**0.5)
      if(v0.GE.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*((v0-beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.GE.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*((v0+beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0-beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0+beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      av0=ABS(v0)
      if(dy.EQ.0.0)then
      etr=-1.0*ak1*(av0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4532  continue
c
C***********************END SIO_2 Y-LOOP********************************
c
c*************SIO_4 Y-LOOP***********************************
      if(ipt(i).EQ.4)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4533
      endif
      drc=z(i)*100.0
      wid=widtc*100.0
      qc=qtc*92903.0
```

```
      uvdir=ATAN2(ycur(iax,iay),xcur(iax,iay))
      ut0=qc/(drc*wid)
      ut=ut0/(((((dx+0.00001)**2+(dy+0.00001)**2)**0.5)**akr_4)
      v0=ycur(iax,iay)+ut*SIN(uvdir)
      uave=(u_blend*ucur+u)/(u_blend+1)
      dxb=uave*per
      deltar=0.72*dxb*(2.5*ruf/dxb)**0.25
      fw=EXP(5.2*((2.5*ruf/dxb)**0.2)-6.0)
      shield=(uave**2/((rhos/rho-1.0)*g*ruf))*fw
      s(i)=srivy4
      ak1=aky1_4
      ak2=ak2_4
      epsilon_l=0.00035*ak2*(uave**0.68)*(shield**0.4)*deltar*g*per
      if(epsilon_l.EQ.0.0000000001)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      surx=-1.0*akz_4*(surdz**2.0)/eps
      decsur=EXP(surx)
      if(dmix(iax,iay).GE.depth(iax,iay).AND.depth(iax,iay).GT.0.0)
     &dmix(iax,iay)=depth(iax,iay)
      if(dmix(iax,iay).LT.surdz)decsur=decsur**thermo
      ps1(i)=s(i)*decsur/((v0**2+4.0*ruf*eps)**0.5)
      if(v0.GE.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*((v0-beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.GE.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*((v0+beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0-beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0+beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      av0=ABS(v0)
      if(dy.EQ.0.0)then
      etr=-1.0*ak1*(av0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4533  continue
c
```

```
C**********************END SIO_4 Y-LOOP********************************
c
c*************SIO Offshore  OUTFALL Y-LOOP***********************************
      if(ipt(i).EQ.5)then
      if(depth(iax,iay).LT.10.0)then
      ps1(i)=0.0
      ps1b(i)=0.0
      etr=0.0
      etrb=0.0
      go to 4534
      endif
      if(ycur(iax,iay).LT.0.0)uwav=-1.0*uwav
      v0=ycur(iax,iay)+uwav
      s(i)=srivy5
      ak1=aky1_5
      ak2=ak2_5
      epsilon_l=0.00035*ak2*(u**0.68)*(shield**0.4)*delta*g*per
      if(epsilon_l.EQ.0.0000000001)epsilon_l=0.0000000001
      eps=epsilon_l+epsilonw
      ps1(i)=s(i)/((u**2+4.0*dl*eps)**0.5)
      if(v0.GE.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*((v0-beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.GE.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*((v0+beta*v0)+((4.0*ruf*eps)**0.5))/(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.LT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0-beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      if(v0.LT.0.0.AND.dy.GT.0.0)then
      etr=-1.0*ak1*(-1.0*(v0+beta*v0)+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      av0=ABS(v0)
      if(dy.EQ.0.0)then
      etr=-1.0*ak1*(av0+((4.0*ruf*eps)**0.5))/
     &(2.0*eps)
      endif
      endif
4534    continue
c
C**********************END SIO Offshore OUTFALL Y-LOOP*******************
```

```
c
1777    continue
        ps3=etr*((dx**2)+(dy**2))**0.5
        ps3b=etrb*((dx**2)+(dy**2))**0.5
c***
        ps2=EXP(ps3)
        ps2b=EXP(ps3b)
        pny(iax,iay)=pny(iax,iay)+(ps1(i)*ps2)
        bry(iax,iay)=bry(iax,iay)+(ps1b(i)*ps2b)
        bryav(iax,iay)=bryav(iax,iay)+(ps1bav(i)*ps2b)
c
1850    continue
1800    continue
1799    continue
c***********END V LOOPS**************************************************
c
c sum pnx and pny together
        do 2800 iax=1,imax
        do 2850 iay=1,jmax
c         if(pnx(iax,iay).LE.0.00000001)pnx(iax,iay)=0.00000001
c         if(pny(iax,iay).LE.0.00000001)pny(iax,iay)=0.00000001
c         if(pnx(iax,iay).GE.1000000000.0)pnx(iax,iay)=1000000000.0
c         if(pny(iax,iay).GE.1000000000.0)pny(iax,iay)=1000000000.0
        pn(iax,iay)=(pnx(iax,iay)+pny(iax,iay))/2.0
c         pn_l(iax,iay)=LOG(pn(iax,iay))
        brn(iax,iay)=(brx(iax,iay)+bry(iax,iay))/2.0
        brnav(iax,iay)=(brxav(iax,iay)+bryav(iax,iay))/2.0
c
c
2850   continue
2800   continue
c
        if(isal.EQ.1)then
c find maximum pnmax
        pnmax=0.0
        brnmax=0.0
        do 2900 iax=1,imax
        do 2950 iay=1,jmax
        if(pn(iax,iay).GT.pnmax)pnmax=pn(iax,iay)
        if(brn(iax,iay).GT.brnmax)brnmax=brn(iax,iay)
2950   continue
2900   continue
        else
```

```
      brnmax=0.0
      do 2902 iax=1,imax
      do 2953 iay=1,jmax
      if(brn(iax,iay).GT.brnmax)brnmax=brn(iax,iay)
2953  continue
2902  continue
      pnmax=pnman
      endif
c
      if(ibrn.EQ.0)brnmax=brnman
      pnmx_l=LOG(pnmax)
      write(*,*)pnmax,brnmax
c
c*****SALINITY LOOP
      do 3100 iax=1,imax
      do 3150 iay=1,jmax
c*salinity at depth increment surdz below surface
      salsur(iax,iay)=salo*(1.0-((pn(iax,iay)/pnmax)**aksal))
      if(salsur(iax,iay).LT.0.0)salsur(iax,iay)=0.0
      if(depth(iax,iay).LE.0.0)salsur(iax,iay)=0.0
c**salinity at elevation botdz above the bottom
      salbrn(iax,iay)=salo*(1-(brn(iax,iay)/brnmax))+brine*
     &(brn(iax,iay)/brnmax)
c**depth averaged brine salinity
      sbrnav(iax,iay)=salo*(1-(brnav(iax,iay)/brnmax))+brine*
     &(brnav(iax,iay)/brnmax)
      if(iax.EQ.150.AND.iay.EQ.92)write(*,*)salbrn(iax,iay)
      if(iax.EQ.150.AND.iay.EQ.92)write(*,*)sbrnav(iax,iay)
c*salinity of sewage at depth increment thrmdz above thermocline
      saldep(iax,iay)=salo*(1.0-((pn(iax,iay)/pnmax)**aksal))
      if(saldep(iax,iay).LT.0.0)saldep(iax,iay)=0.0
      if(depth(iax,iay).LE.0.0)saldep(iax,iay)=0.0
c
c SIO_1 Inshore
      if(iax.EQ.174.AND.iay.EQ.94)write(*,*)iax,iay,pn(iax,iay),
     &salsur(iax,iay),salbrn(iax,iay),dmix(iax,iay)
c upper left (NW) corner of grid
      if(iax.EQ.1.AND.iay.EQ.1)write(*,*)iax,iay,pn(iax,iay),
     &salsur(iax,iay),salbrn(iax,iay),dmix(iax,iay)
c SIO_3 Inshore
      if(iax.EQ.173.AND.iay.EQ.96)write(*,*)iax,iay,pn(iax,iay),
     &salsur(iax,iay),salbrn(iax,iay),dmix(iax,iay)
c SIO_2
```

```
      if(iax.EQ.193.AND.iay.EQ.100)write(*,*)iax,iay,pn(iax,iay),
     &salsur(iax,iay),salbrn(iax,iay),dmix(iax,iay)
c SIO_4 Outfall
      if(iax.EQ.204.AND.iay.EQ.109)write(*,*)iax,iay,pn(iax,iay),
     &salsur(iax,iay),salbrn(iax,iay),dmix(iax,iay)
3150  continue
3100  continue
c
C****SALINITY MAXIMUM AND MINIMUM LOOP
      salsmx=0.0
      saldmx=0.0
      salbmn=33.52
      do 3160 iax=1,imax
      do 3170 iay=1,jmax
      if(salsur(iax,iay).GT.salsmx)salsmx=salsur(iax,iay)
      if(saldep(iax,iay).GT.saldmx)saldmx=saldep(iax,iay)
      if(salbrn(iax,iay).LT.salbmn)salbmn=salbrn(iax,iay)
3170  continue
3160  continue
      write(*,*)salsmx,saldmx,salbmn
c
c****DILUTION LOOP
      do 3180 iax=1,imax
      do 3190 iay=1,jmax
c dilution of storm water sources at depth increment surdz below surface
      dilriv(iax,iay)=LOG10(salsmx/(salsmx-salsur(iax,iay)+back))
c**storm waterdilution at elevation botdz above the bottom
     dilbrn(iax,iay)=LOG10((rawbrn-salbmn)/
     &(salbrn(iax,iay)-salbmn+back))
c**depth averaged brine dilution
     dbrnav(iax,iay)=LOG10((rawbrn-salbmn)/
     &(sbrnav(iax,iay)-salbmn+back))
      if(iax.EQ.150.AND.iay.EQ.92)write(*,*)dilbrn(iax,iay)
      if(iax.EQ.150.AND.iay.EQ.92)write(*,*)dbrnav(iax,iay)
c dilution of storm water at depth increment thrmdz above thermocline
      dildep(iax,iay)=LOG10(saldmx/(saldmx-saldep(iax,iay)+back))
c TSS counts of storm water at depth thrmdz above thermocline
      col(iax,iay)=LOG10(coli)-dildep(iax,iay)
      if(col(iax,iay).LT.0.0)col(iax,iay)=0.0
c
c TSS counts of outfall source at depth surdz below surface
      colriv(iax,iay)=LOG10(coli)-dilriv(iax,iay)
      if(colriv(iax,iay).LT.0.0)colriv(iax,iay)=0.0
```

```
c
3190    continue
3180    continue
c
c
        do 901 iay=1,jmax
        write(21,999)(pn(iax,iay),iax=1,imax)
901     continue
c
        close(21)
c
        do 902 iay=1,jmax
        write(22,999)(pn_l(iax,iay),iax=1,imax)
902     continue
c
        close(22)
c
c
        do 888 iax=1,imax
        do 887 iay=1,jmax
        if(depth(iax,iay).LT.surdz)salsur(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)salsur(iax,iay)=-2.0
        if(depth(iax,iay).LT.botdz)salbrn(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)salbrn(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)dilbrn(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)dildep(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)saldep(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)col(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)colriv(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)sbrnav(iax,iay)=-2.0
        if(depth(iax,iay).LE.0.0)dbrnav(iax,iay)=-2.0
887     continue
888     continue
c
        do 903 iay=1,jmax
        write(23,995)(salsur(iax,iay),iax=1,imax)
903     continue
c
        close(23)
c
c
        do 905 iay=1,jmax
        write(25,995)(salbrn(iax,iay),iax=1,imax)
```

```
905    continue
c
      close(25)
c
c
      do 906 iay=1,jmax
      write(26,995)(dilbrn(iax,iay),iax=1,imax)
906    continue
c
      close(26)
c
      do 904 iay=1,jmax
      write(24,999)(dilriv(iax,iay),iax=1,imax)
904    continue
c
      close(24)
cc
      do 907 iay=1,jmax
      write(27,995)(saldep(iax,iay),iax=1,imax)
907    continue
c
      close(27)
c
c
      do 908 iay=1,jmax
      write(28,995)(dildep(iax,iay),iax=1,imax)
908    continue
c
      close(28)
c
c
      do 909 iay=1,jmax
      write(29,995)(col(iax,iay),iax=1,imax)
909    continue
c
      close(29)
c
      do 910 iay=1,jmax
      write(30,995)(dmix(iax,iay),iax=1,imax)
910    continue
c
      close(30)
c
```

```
      do 911 iay=1,jmax
      write(31,995)(colriv(iax,iay),iax=1,imax)
911   continue
c
      close(31)
c
c
      do 912 iay=1,jmax
      write(32,995)(sbrnav(iax,iay),iax=1,imax)
912   continue
c
      close(32)
c
c
      do 913 iay=1,jmax
      write(33,995)(dbrnav(iax,iay),iax=1,imax)
913   continue
c
      close(33)
c
999   format(200e12.3)
995   format(200e15.6)
c
      stop
      end
```

APPENDIX G

AMEC MONITORING DATA

Appendix G:
AMEC Monitoring Data

Station Code	Sample Date	Time		Units	Result	Collection Device Name	Agency
906SDL062OD	03/Nov/11	8:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	9:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	9:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	9:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	9:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	10:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	10:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	10:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	10:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	11:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	11:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	11:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	11:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	12:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	12:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	12:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	12:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	13:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	13:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	13:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	13:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	14:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	14:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	14:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	14:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	15:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	15:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	15:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	15:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	16:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	16:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	16:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	16:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	17:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	17:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	17:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	17:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	18:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	18:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	18:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	18:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	19:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	19:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	19:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	19:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	20:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	20:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	20:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	20:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	21:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	21:15	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	21:30	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	21:45	PST	inches	0	Hach tipping bucket rain gauge	AMEC
906SDL062OD	03/Nov/11	22:00	PST	inches	0	Hach tipping bucket rain gauge	AMEC

Station Code	Sample Date	Time	Agency	Location	Result		Collection Device Name
906SDL062OD	20/Nov/11	14:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	14:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	14:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	14:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	15:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	15:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	15:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	15:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	16:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	16:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	16:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	16:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	17:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	17:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	17:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	17:45	AMEC	Pipe	1.29	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	18:00	AMEC	Pipe	1.76	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	18:15	AMEC	Pipe	5.48	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	18:30	AMEC	Pipe	8.77	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	18:45	AMEC	Pipe	9.44	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	19:00	AMEC	Pipe	10.44	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	19:15	AMEC	Pipe	9.33	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	19:30	AMEC	Pipe	6.54	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	19:45	AMEC	Pipe	4.83	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	20:00	AMEC	Pipe	4.45	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	20:15	AMEC	Pipe	4.05	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	20:30	AMEC	Pipe	3.62	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	20:45	AMEC	Pipe	3.7	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	21:00	AMEC	Pipe	4.5	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	21:15	AMEC	Pipe	3.67	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	21:30	AMEC	Pipe	3.52	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	21:45	AMEC	Pipe	3.39	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	22:00	AMEC	Pipe	3.14	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	22:15	AMEC	Pipe	2.63	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	22:30	AMEC	Pipe	2.09	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	22:45	AMEC	Pipe	1.66	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	23:00	AMEC	Pipe	1.39	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	23:15	AMEC	Pipe	1.32	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	23:30	AMEC	Pipe	1.62	cfs	Hach 950 Flow Meter
906SDL062OD	20/Nov/11	23:45	AMEC	Pipe	4.59	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	0:00	AMEC	Pipe	6.12	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	0:15	AMEC	Pipe	3.99	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	0:30	AMEC	Pipe	3.16	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	0:45	AMEC	Pipe	3.12	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	1:00	AMEC	Pipe	2.75	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	1:15	AMEC	Pipe	2.15	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	1:30	AMEC	Pipe	1.59	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	1:45	AMEC	Pipe	1.11	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	2:00	AMEC	Pipe	0.761	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	2:15	AMEC	Pipe	0.53	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	2:30	AMEC	Pipe	0.339	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	2:45	AMEC	Pipe	0.249	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	3:00	AMEC	Pipe	0.204	cfs	Hach 950 Flow Meter
906SDL062OD	21/Nov/11	3:15	AMEC	Pipe	0.146	cfs	Hach 950 Flow Meter

Station Code	Sample Date	Time	Agency	Location	Result	Units	Collection Device Name
906SDL062OD	07/Feb/12	14:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	14:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	14:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	14:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	15:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	15:15	AMEC	Pipe	0.01	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	15:30	AMEC	Pipe	0.095	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	15:45	AMEC	Pipe	0.28	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	16:00	AMEC	Pipe	1.9	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	16:15	AMEC	Pipe	2.2	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	16:30	AMEC	Pipe	1.33	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	16:45	AMEC	Pipe	2.23	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	17:00	AMEC	Pipe	4.44	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	17:15	AMEC	Pipe	6.56	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	17:30	AMEC	Pipe	5.14	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	17:45	AMEC	Pipe	2.47	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	18:00	AMEC	Pipe	2.04	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	18:15	AMEC	Pipe	1.02	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	18:30	AMEC	Pipe	0.293	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	18:45	AMEC	Pipe	0.047	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	19:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	19:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	19:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	19:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	20:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	20:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	20:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	20:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	07/Feb/12	21:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter

Station Code	Sample Date	Time	Agency	Location	Result	Units	Collection Device Name
906SDL062OD	16/Mar/12	17:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	18:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	18:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	18:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	18:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	19:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	19:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	19:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	19:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	20:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	20:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	20:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	20:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	21:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	21:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	21:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	21:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	22:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	22:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	22:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	22:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	23:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	23:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	23:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	16/Mar/12	23:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	0:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	0:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	0:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	0:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	1:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	1:15	AMEC	Pipe	0.053	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	1:30	AMEC	Pipe	0.428	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	1:45	AMEC	Pipe	0.252	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	2:00	AMEC	Pipe	0.065	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	2:15	AMEC	Pipe	0.011	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	2:30	AMEC	Pipe	0.031	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	2:45	AMEC	Pipe	0.031	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	3:00	AMEC	Pipe	0.017	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	3:15	AMEC	Pipe	0.017	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	3:30	AMEC	Pipe	0.014	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	3:45	AMEC	Pipe	0.01	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	4:00	AMEC	Pipe	0.01	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	4:15	AMEC	Pipe	0.225	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	4:30	AMEC	Pipe	0.44	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	4:45	AMEC	Pipe	0.361	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	5:00	AMEC	Pipe	0.493	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	5:15	AMEC	Pipe	1.74	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	5:30	AMEC	Pipe	1.68	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	5:45	AMEC	Pipe	2.23	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	6:00	AMEC	Pipe	2.18	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	6:15	AMEC	Pipe	4.25	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	6:30	AMEC	Pipe	1.62	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	6:45	AMEC	Pipe	1.78	cfs	Hach 950 Flow Meter
906SDL062OD	17/Mar/12	7:00	AMEC	Pipe	2.13	cfs	Hach 950 Flow Meter

Station Code	Sample Date	Time	Agency	Location	Result	Units	Collection Device Name
906SDL157OD	20/Nov/11	9:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	9:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	9:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	10:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	10:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	10:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	10:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	11:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	11:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	11:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	11:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	12:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	12:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	12:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	12:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	13:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	13:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	13:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	13:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	14:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	14:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	14:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	14:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	15:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	15:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	15:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	15:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	16:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	16:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	16:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	16:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	17:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	17:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	17:30	AMEC	Pipe	0.003	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	17:45	AMEC	Pipe	0.56	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	18:00	AMEC	Pipe	1.26	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	18:15	AMEC	Pipe	1.25	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	18:30	AMEC	Pipe	2.73	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	18:45	AMEC	Pipe	2.96	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	19:00	AMEC	Pipe	3.22	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	19:15	AMEC	Pipe	2.26	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	19:30	AMEC	Pipe	1.76	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	19:45	AMEC	Pipe	1.49	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	20:00	AMEC	Pipe	0.96	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	20:15	AMEC	Pipe	0.94	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	20:30	AMEC	Pipe	0.92	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	20:45	AMEC	Pipe	1.72	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	21:00	AMEC	Pipe	1.68	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	21:15	AMEC	Pipe	1.08	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	21:30	AMEC	Pipe	0.93	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	21:45	AMEC	Pipe	0.97	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	22:00	AMEC	Pipe	0.95	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	22:15	AMEC	Pipe	0.84	cfs	Hach 950 Flow Meter
906SDL157OD	20/Nov/11	22:30	AMEC	Pipe	0.75	cfs	Hach 950 Flow Meter

Station Code	Sample Date	Time	Agency	Location	Result	Units	Collection Device Name
906SDL157OD	07/Feb/12	14:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	14:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	14:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	14:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	15:00	AMEC	Pipe	0.001	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	15:15	AMEC	Pipe	0.11	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	15:30	AMEC	Pipe	0.13	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	15:45	AMEC	Pipe	0.16	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	16:00	AMEC	Pipe	0.45	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	16:15	AMEC	Pipe	1.31	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	16:30	AMEC	Pipe	1.34	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	16:45	AMEC	Pipe	2.52	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	17:00	AMEC	Pipe	2.88	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	17:15	AMEC	Pipe	3.48	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	17:30	AMEC	Pipe	3.15	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	17:45	AMEC	Pipe	3.17	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	18:00	AMEC	Pipe	2.73	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	18:15	AMEC	Pipe	2.07	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	18:30	AMEC	Pipe	1.64	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	18:45	AMEC	Pipe	1.27	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	19:00	AMEC	Pipe	0.77	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	19:15	AMEC	Pipe	0.51	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	19:30	AMEC	Pipe	0.24	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	19:45	AMEC	Pipe	0.1	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	20:00	AMEC	Pipe	0.022	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	20:15	AMEC	Pipe	0.003	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	20:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	20:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	07/Feb/12	21:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter

Station Code	Sampl eDate	Time	Agency	Location	Result	Units	Collection Device Name
906SDL157OD	16/Mar/12	17:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	18:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	18:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	18:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	18:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	19:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	19:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	19:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	19:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	20:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	20:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	20:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	20:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	21:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	21:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	21:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	21:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	22:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	22:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	22:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	22:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	23:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	23:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	23:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	16/Mar/12	23:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	0:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	0:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	0:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	0:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	1:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	1:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	1:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	1:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	2:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	2:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	2:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	2:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	3:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	3:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	3:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	3:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	4:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	4:15	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	4:30	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	4:45	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	5:00	AMEC	Pipe	0	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	5:15	AMEC	Pipe	0.001	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	5:30	AMEC	Pipe	0.045	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	5:45	AMEC	Pipe	0.19	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	6:00	AMEC	Pipe	0.51	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	6:15	AMEC	Pipe	0.9	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	6:30	AMEC	Pipe	1.14	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	6:45	AMEC	Pipe	0.78	cfs	Hach 950 Flow Meter
906SDL157OD	17/Mar/12	7:00	AMEC	Pipe	0.39	cfs	Hach 950 Flow Meter

Station Code	Date	Time	Result	Units	TSS (mg/L)	TSS Load (kg/min)
906SDL062OD	20/Nov/11	14:00	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	14:15	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	14:30	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	14:45	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	15:00	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	15:15	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	15:30	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	15:45	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	16:00	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	16:15	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	16:30	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	16:45	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	17:00	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	17:15	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	17:30	0	cfs	260	0.0000
906SDL062OD	20/Nov/11	17:45	1.29	cfs	260	0.5698
906SDL062OD	20/Nov/11	18:00	1.76	cfs	260	0.7775
906SDL062OD	20/Nov/11	18:15	5.48	cfs	260	2.4208
906SDL062OD	20/Nov/11	18:30	8.77	cfs	260	3.8741
906SDL062OD	20/Nov/11	18:45	9.44	cfs	260	4.1701
906SDL062OD	20/Nov/11	19:00	10.44	cfs	260	4.6118
906SDL062OD	20/Nov/11	19:15	9.33	cfs	260	4.1215
906SDL062OD	20/Nov/11	19:30	6.54	cfs	260	2.8890
906SDL062OD	20/Nov/11	19:45	4.83	cfs	260	2.1336
906SDL062OD	20/Nov/11	20:00	4.45	cfs	260	1.9658
906SDL062OD	20/Nov/11	20:15	4.05	cfs	260	1.7891
906SDL062OD	20/Nov/11	20:30	3.62	cfs	260	1.5991
906SDL062OD	20/Nov/11	20:45	3.7	cfs	260	1.6344
906SDL062OD	20/Nov/11	21:00	4.5	cfs	260	1.9878
906SDL062OD	20/Nov/11	21:15	3.67	cfs	260	1.6212
906SDL062OD	20/Nov/11	21:30	3.52	cfs	260	1.5549
906SDL062OD	20/Nov/11	21:45	3.39	cfs	260	1.4975
906SDL062OD	20/Nov/11	22:00	3.14	cfs	260	1.3871
906SDL062OD	20/Nov/11	22:15	2.63	cfs	260	1.1618
906SDL062OD	20/Nov/11	22:30	2.09	cfs	260	0.9232
906SDL062OD	20/Nov/11	22:45	1.66	cfs	260	0.7333
906SDL062OD	20/Nov/11	23:00	1.39	cfs	260	0.6140
906SDL062OD	20/Nov/11	23:15	1.32	cfs	260	0.5831
906SDL062OD	20/Nov/11	23:30	1.62	cfs	260	0.7156
906SDL062OD	20/Nov/11	23:45	4.59	cfs	260	2.0276
906SDL062OD	21/Nov/11	0:00	6.12	cfs	260	2.7035
906SDL062OD	21/Nov/11	0:15	3.99	cfs	260	1.7626
906SDL062OD	21/Nov/11	0:30	3.16	cfs	260	1.3959
906SDL062OD	21/Nov/11	0:45	3.12	cfs	260	1.3782
906SDL062OD	21/Nov/11	1:00	2.75	cfs	260	1.2148
906SDL062OD	21/Nov/11	1:15	2.15	cfs	260	0.9497
906SDL062OD	21/Nov/11	1:30	1.59	cfs	260	0.7024
906SDL062OD	21/Nov/11	1:45	1.11	cfs	260	0.4903
906SDL062OD	21/Nov/11	2:00	0.761	cfs	260	0.3362
906SDL062OD	21/Nov/11	2:15	0.53	cfs	260	0.2341
906SDL062OD	21/Nov/11	2:30	0.339	cfs	260	0.1498
906SDL062OD	21/Nov/11	2:45	0.249	cfs	260	0.1100
906SDL062OD	21/Nov/11	3:00	0.204	cfs	260	0.0901

Station Code	Date	Time	Result	Units	TSS (mg/L)	TSS Load (kg/min)
906SDL062OD	07/Feb/12	14:00	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	14:15	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	14:30	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	14:45	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	15:00	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	15:15	0.01	cfs	130	0.0022
906SDL062OD	07/Feb/12	15:30	0.095	cfs	130	0.0210
906SDL062OD	07/Feb/12	15:45	0.28	cfs	130	0.0618
906SDL062OD	07/Feb/12	16:00	1.9	cfs	130	0.4197
906SDL062OD	07/Feb/12	16:15	2.2	cfs	130	0.4859
906SDL062OD	07/Feb/12	16:30	1.33	cfs	130	0.2938
906SDL062OD	07/Feb/12	16:45	2.23	cfs	130	0.4925
906SDL062OD	07/Feb/12	17:00	4.44	cfs	130	0.9807
906SDL062OD	07/Feb/12	17:15	6.56	cfs	130	1.4489
906SDL062OD	07/Feb/12	17:30	5.14	cfs	130	1.1353
906SDL062OD	07/Feb/12	17:45	2.47	cfs	130	0.5456
906SDL062OD	07/Feb/12	18:00	2.04	cfs	130	0.4506
906SDL062OD	07/Feb/12	18:15	1.02	cfs	130	0.2253
906SDL062OD	07/Feb/12	18:30	0.293	cfs	130	0.0647
906SDL062OD	07/Feb/12	18:45	0.047	cfs	130	0.0104
906SDL062OD	07/Feb/12	19:00	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	19:15	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	19:30	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	19:45	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	20:00	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	20:15	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	20:30	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	20:45	0	cfs	130	0.0000
906SDL062OD	07/Feb/12	21:00	0	cfs	130	0.0000
						99.57
			27,050		130	**99.57**
			Total Flow (cf)			**Total Load (kg/event)**

Station Code	Date	Time	Result	Units	TSS (mg/L)	TSS Load (kg/min)
906SDL062OD	16/Mar/12	17:45	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	18:00	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	18:15	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	18:30	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	18:45	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	19:00	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	19:15	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	19:30	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	19:45	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	20:00	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	20:15	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	20:30	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	20:45	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	21:00	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	21:15	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	21:30	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	21:45	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	22:00	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	22:15	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	22:30	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	22:45	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	23:00	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	23:15	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	23:30	0	cfs	81	0.0000
906SDL062OD	16/Mar/12	23:45	0	cfs	81	0.0000
906SDL062OD	17/Mar/12	0:00	0	cfs	81	0.0000
906SDL062OD	17/Mar/12	0:15	0	cfs	81	0.0000
906SDL062OD	17/Mar/12	0:30	0	cfs	81	0.0000
906SDL062OD	17/Mar/12	0:45	0	cfs	81	0.0000
906SDL062OD	17/Mar/12	1:00	0	cfs	81	0.0000
906SDL062OD	17/Mar/12	1:15	0.053	cfs	81	0.0073
906SDL062OD	17/Mar/12	1:30	0.428	cfs	81	0.0589
906SDL062OD	17/Mar/12	1:45	0.252	cfs	81	0.0347
906SDL062OD	17/Mar/12	2:00	0.065	cfs	81	0.0089
906SDL062OD	17/Mar/12	2:15	0.011	cfs	81	0.0015
906SDL062OD	17/Mar/12	2:30	0.031	cfs	81	0.0043
906SDL062OD	17/Mar/12	2:45	0.031	cfs	81	0.0043
906SDL062OD	17/Mar/12	3:00	0.017	cfs	81	0.0023
906SDL062OD	17/Mar/12	3:15	0.017	cfs	81	0.0023
906SDL062OD	17/Mar/12	3:30	0.014	cfs	81	0.0019
906SDL062OD	17/Mar/12	3:45	0.01	cfs	81	0.0014
906SDL062OD	17/Mar/12	4:00	0.01	cfs	81	0.0014
906SDL062OD	17/Mar/12	4:15	0.225	cfs	81	0.0310
906SDL062OD	17/Mar/12	4:30	0.44	cfs	81	0.0606
906SDL062OD	17/Mar/12	4:45	0.361	cfs	81	0.0497
906SDL062OD	17/Mar/12	5:00	0.493	cfs	81	0.0678
906SDL062OD	17/Mar/12	5:15	1.74	cfs	81	0.2395
906SDL062OD	17/Mar/12	5:30	1.68	cfs	81	0.2312
906SDL062OD	17/Mar/12	5:45	2.23	cfs	81	0.3069
906SDL062OD	17/Mar/12	6:00	2.18	cfs	81	0.3000
906SDL062OD	17/Mar/12	6:15	4.25	cfs	81	0.5849
906SDL062OD	17/Mar/12	6:30	1.62	cfs	81	0.2229
906SDL062OD	17/Mar/12	6:45	1.78	cfs	81	0.2450
906SDL062OD	17/Mar/12	7:00	2.13	cfs	81	0.2931

Station Code	Date	Time	Result	Units	TSS (mg/L)	TSS Load (kg/min)
906SDL157OD	20/Nov/11	9:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	9:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	9:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	10:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	10:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	10:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	10:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	11:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	11:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	11:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	11:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	12:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	12:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	12:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	12:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	13:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	13:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	13:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	13:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	14:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	14:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	14:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	14:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	15:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	15:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	15:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	15:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	16:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	16:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	16:30	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	16:45	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	17:00	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	17:15	0	cfs	340	0.0000
906SDL157OD	20/Nov/11	17:30	0.003	cfs	340	0.0017
906SDL157OD	20/Nov/11	17:45	0.56	cfs	340	0.3235
906SDL157OD	20/Nov/11	18:00	1.26	cfs	340	0.7279
906SDL157OD	20/Nov/11	18:15	1.25	cfs	340	0.7221
906SDL157OD	20/Nov/11	18:30	2.73	cfs	340	1.5770
906SDL157OD	20/Nov/11	18:45	2.96	cfs	340	1.7099
906SDL157OD	20/Nov/11	19:00	3.22	cfs	340	1.8601
906SDL157OD	20/Nov/11	19:15	2.26	cfs	340	1.3055
906SDL157OD	20/Nov/11	19:30	1.76	cfs	340	1.0167
906SDL157OD	20/Nov/11	19:45	1.49	cfs	340	0.8607
906SDL157OD	20/Nov/11	20:00	0.96	cfs	340	0.5546
906SDL157OD	20/Nov/11	20:15	0.94	cfs	340	0.5430
906SDL157OD	20/Nov/11	20:30	0.92	cfs	340	0.5315
906SDL157OD	20/Nov/11	20:45	1.72	cfs	340	0.9936
906SDL157OD	20/Nov/11	21:00	1.68	cfs	340	0.9705
906SDL157OD	20/Nov/11	21:15	1.08	cfs	340	0.6239
906SDL157OD	20/Nov/11	21:30	0.93	cfs	340	0.5372
906SDL157OD	20/Nov/11	21:45	0.97	cfs	340	0.5603
906SDL157OD	20/Nov/11	22:00	0.95	cfs	340	0.5488
906SDL157OD	20/Nov/11	22:15	0.84	cfs	340	0.4852

Station Code	Date	Time	Result	Units	TSS (mg/L)	TSS Load (kg/min)
906SDL157OD	07/Feb/12	14:00	0	cfs	230	0.0000
906SDL157OD	07/Feb/12	14:15	0	cfs	230	0.0000
906SDL157OD	07/Feb/12	14:30	0	cfs	230	0.0000
906SDL157OD	07/Feb/12	14:45	0	cfs	230	0.0000
906SDL157OD	07/Feb/12	15:00	0.001	cfs	230	0.0004
906SDL157OD	07/Feb/12	15:15	0.11	cfs	230	0.0430
906SDL157OD	07/Feb/12	15:30	0.13	cfs	230	0.0508
906SDL157OD	07/Feb/12	15:45	0.16	cfs	230	0.0625
906SDL157OD	07/Feb/12	16:00	0.45	cfs	230	0.1758
906SDL157OD	07/Feb/12	16:15	1.31	cfs	230	0.5119
906SDL157OD	07/Feb/12	16:30	1.34	cfs	230	0.5236
906SDL157OD	07/Feb/12	16:45	2.52	cfs	230	0.9847
906SDL157OD	07/Feb/12	17:00	2.88	cfs	230	1.1254
906SDL157OD	07/Feb/12	17:15	3.48	cfs	230	1.3599
906SDL157OD	07/Feb/12	17:30	3.15	cfs	230	1.2309
906SDL157OD	07/Feb/12	17:45	3.17	cfs	230	1.2387
906SDL157OD	07/Feb/12	18:00	2.73	cfs	230	1.0668
906SDL157OD	07/Feb/12	18:15	2.07	cfs	230	0.8089
906SDL157OD	07/Feb/12	18:30	1.64	cfs	230	0.6409
906SDL157OD	07/Feb/12	18:45	1.27	cfs	230	0.4963
906SDL157OD	07/Feb/12	19:00	0.77	cfs	230	0.3009
906SDL157OD	07/Feb/12	19:15	0.51	cfs	230	0.1993
906SDL157OD	07/Feb/12	19:30	0.24	cfs	230	0.0938
906SDL157OD	07/Feb/12	19:45	0.1	cfs	230	0.0391
906SDL157OD	07/Feb/12	20:00	0.022	cfs	230	0.0086
906SDL157OD	07/Feb/12	20:15	0.003	cfs	230	0.0012
906SDL157OD	07/Feb/12	20:30	0	cfs	230	0.0000
906SDL157OD	07/Feb/12	20:45	0	cfs	230	0.0000
906SDL157OD	07/Feb/12	21:00	0	cfs	230	0.0000
						164.45
			25,250		230	**164.45**
			Total Flow (cf)			**Total Load (kg/event)**

Station Code	Date	Time	Result	Units	TSS (mg/L)	TSS Load (kg/min)
906SDL157OD	16/Mar/12	17:45	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	18:00	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	18:15	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	18:30	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	18:45	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	19:00	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	19:15	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	19:30	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	19:45	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	20:00	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	20:15	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	20:30	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	20:45	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	21:00	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	21:15	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	21:30	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	21:45	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	22:00	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	22:15	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	22:30	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	22:45	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	23:00	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	23:15	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	23:30	0	cfs	170	0.0000
906SDL157OD	16/Mar/12	23:45	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	0:00	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	0:15	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	0:30	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	0:45	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	1:00	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	1:15	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	1:30	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	1:45	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	2:00	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	2:15	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	2:30	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	2:45	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	3:00	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	3:15	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	3:30	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	3:45	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	4:00	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	4:15	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	4:30	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	4:45	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	5:00	0	cfs	170	0.0000
906SDL157OD	17/Mar/12	5:15	0.001	cfs	170	0.0003
906SDL157OD	17/Mar/12	5:30	0.045	cfs	170	0.0130
906SDL157OD	17/Mar/12	5:45	0.19	cfs	170	0.0549
906SDL157OD	17/Mar/12	6:00	0.51	cfs	170	0.1473
906SDL157OD	17/Mar/12	6:15	0.9	cfs	170	0.2599
906SDL157OD	17/Mar/12	6:30	1.14	cfs	170	0.3293
906SDL157OD	17/Mar/12	6:45	0.78	cfs	170	0.2253
906SDL157OD	17/Mar/12	7:00	0.39	cfs	170	0.1126