Appendix A
Data Dictionary

Coverages *

Coverage	Feature	Class	Attribute	Value	Description
alb5900	USGS gauge station 8075900	Point	usgs_stat	varies	USGS gauge station number
			location	varies	Physical location as described by USGS
			drain_area	varies	USGS drainage area (mi^2)
			per_of_rec	varies	Available per. of record for flow data
			msrd_run	varies	Avg. msrd runoff depth (1961-1990)
bcseg	Tributaries entering the Upper Houston Ship Channel	Arc	none		
bodgr30	Visualization of areal BOD loading over the Upper HSC watershed (only shows polygons greater than 30000 m^2 -- cannot be used for modeling purposes)	Poly.	grid-code	varies	Areal BOD loading (kg/yr/ha)
cvemc30	Visualization of EMC distribution over the Upper HSC watershed (only shows polygons greater than 30000 m^2 -- cannot be used for modeling purposes)	Poly.	grid-code	0	EMC for water
				4	EMC for agriculture
				6	EMC for open, wetlands, and water
				9	EMC for high density urban
				13	EMC for barren
				15	EMC for residential
cvnoedit	Watershed area delineated without DLG "burn"	Arc	none		
cvpsmsrd	38 point source dischargers along the Upper HSC shoreline which have BOD measurments from Armstrong and Ward (1994)	Point	tnrcc_per_num	varies	TNRCC water quality permit number
			x-coord	varies	x location, in albers units
			y-coord	varies	y location in albers units
cvsegshd	Eight subwatersheds for each water quality segment in the Upper HSC	Poly.	grid-code	1 to 8	Corresponding water quality segment which drains this area
cvshd590	Subwatershed for USGS gauge 8075900	Poly.	grid-code	8075900	USGS gauge which drains this area
dlgedit	Final, edited digital line graph used for digital elevation model "stream burn"	Arc	none		
dlgnohsc	Digital line graph without the HSC defined as a center line	Arc	none		
dlgshd	DLG coverage encompassed by the Upper HSC watershed	Arc	none		

Coverage	Feature	Class	Attribute	Value	Description
gagesd10	All 10 subwatersheds of the USGS gauge stations used in this study	Poly.	grid-code	varies	USGS gauge which drains the corresponding area
gageshd	Each subwatershed for each of the nine USGS gages used in the rainfall/flow and rainfall/runoff correlations	Poly.	grid-code	varies	USGS gauge which drains the corresponding area
gbaydlg	Original 1:100,000 digital line graph of the Galveston Bay area	Arc	none		
gbyhyseg	Geographic projection of the Galveston Bay hydrogaphic segmentation as given in the GBNEP Report-22 (1992)	Poly.	segment-id	varies	Segment # Ward and Armstrong (1992)
			x-coord	varies	Longitude of label point in decimal deg.
			y-coord	varies	Latitude of label point in decimal deg.
Gbhysgpr	Alber projection of gbyhyseg		segment-id	Same as for gbyhyseg	
			x-coord	varies	x location, in albers units
			y-coord	varies	y location in albers units
hscalone	Arc of the centerline used for the HSC definition	Arc	none		
hsclu	Final landuse coverage used for EMC and Runoff Coefficent values. Combines, primarily 1990 data w/ missing areas of ws filled in by 1980 data	Poly.	grid-code	0	Background
				1	Urban
				2	Open
				3	Agriculture
				4	Barren
				5	Wetlands
				6	Residential
				7	Water
				8	Forest
			curve_num	97	CN for grid-code = 1
				80	CN for grid-code = 2
				81	CN for grid-code = 3
				89	CN for grid-code = 4
				67	CN for grid-code = 5
				87	CN for grid-code = 6
				100	CN for grid-code = 7
				77	CN for grid-code = 8

Coverage	Feature	Class	Attribute	Value	Description
			runoff_coeff	0.89	c for grid-code = 1
				0.22	c for grid-code = 2
				0.24	c for grid-code = 3
				0.22	c for grid-code = 4
				0.8	c for grid-code = 5
				0.34	c for grid-code = 6
				1	c for grid-code = 7
				0.15	c for grid-code = 8
hsc_pscv	68 point source dischargers which lie directly on the Upper HSC shoreline	Point	tnrcc_per_num	varies	TNRCC water quality permit number
			x-coord	varies	x location, in albers units
			y-coord	varies	y location in albers units
lu_80dis	Dissolved (no boundary lines) 1980 land use with Anderson Classification	Poly.	lucode	11	Residential
				12	Commerical Services
				13	Industrial
				14	Transportation, Communications
				15	Industrial and Commercial
				16	Mixed Urban or Built-Up Land
				17	Other Urban or Built-Up Land
				21	Cropland and Pasture
				22	Orchards, Groves, Vineyards, Nurseries
				23	Confined Feeding Operations
				31	Herbaceous Rangeland
				32	Shrub and Brush Rangeland
				33	Mixed Rangeland
				41	Deciduous Forest Land
				42	Evergreen Forest Land
				43	Mixed Forest Land
				51	Streams and Canals
				52	Lakes
				53	Reserviors
				54	Bays and Esturaries

Coverage	Feature	Class	Attribute	Value	Description
				61	Forested Wetlands
				62	Nonforested Wetlands
				71	Dry Salt Flats
				72	Beaches
				73	Sandy Areas other than Beaches
				74	Bare Exposed Rock
				75	Strip Mines, Quarries, and Gravel Pits
				76	Transitional Area
				77	Mixed Barren Land
			grid-code	Same as grid-code for hsclu	
lu_80ed	1980 land use of the areas within the hsc watershed which were missing the 1990 data	Poly.	lucode	Same as lucode for lu_80dis	
			grid-code	Same as grid-code for hsclu	
lu_90dis	1990 land use coverage of watershed area as defined by Newell et al. (1992)	Poly.	grid-code	Same as grid-code for hsclu	
lu_90shd	1990 land use coverage of watershed area defined by delineation (mapjoined w/ lu_80ed to get hsclu)	Poly.	grid-code	Same as grid-code for hsclu	
p1cov	Coverage of January rainfall depth grid	Poly.	grid-code	varies	Avg. precip. depth for Jan, 1961-1990
panncov	Coverage of annual rainfall depth grid	Poly.	grid-code	varies	Avg. annual precip. depth, 1961-1990
quadpr	Coverage of four USGS quads encompassing the study area	Poly.		none	
ranncv30	Visualiztion of runoff distribution over the Upper HSC watershed (only shows polygons greater than 30000 m^2 -- can not be used for modeling purposes)	Poly.	grid-code	varies	Areal calculated runoff depth (mm/yr/ha) calculated using rainfall/ runoff/urbanization correlation
rstrcv	Calculated runoff depth from correlation, routed down the streams	Arc	grid-code	varies	Calculated runoff depth using rainfall/ runoff/urbanization eqn. (mm/yr/ha)
segarc	Eight water quality segments	Arc	grid-code	1 to 8	Water quality segment # for model
stat10	All 10 USGS gauge stations used in this study	Point	none		

Coverage	Feature	Class	Attribute	Value	Description
stat_alb	71 USGS gauging stations within Harris, Waller, and Fort Bend Counties	Point	usgs_stat	varies	USGS gauge #
			x-coord	varies	x location, in albers units
			y-coord	varies	y location in albers units
stat_hsc	37 USGS gauging stations in HSC watershed		usgs_stat	varies	USGS gauge #
			x-coord	varies	longitude in decimal degrees
			y-coord	varies	latitude in decimal degrees
stendalb	Start and end points of segmentation for the Upper HSC (Turning Basin and San Jacinto Monument) as read from 1:24,000 USGS quad sheet	Point	x-coord	varies	x location, in albers units
			y-coord	varies	y location in albers units
strnoedit	Stream coverage, delineated without DLG "burn" in	Arc	none		
strshd	Delineated streams (with burn) in the Upper HSC watershed	Arc	none		
str_500	Final coverage of the stream delineation on the 500 cell threshold	Arc	none		
studstat	Coverage of the nine USGS gauges used for this study	Point	usgs_stat	varies	USGS gauge station number
			location	varies	Physical location as described by USGS
			drain_area	varies	USGS drainage area (mi^2)
			per_of_rec	varies	Available per. of record for flow data
			msrd_run	varies	Avg. msrd runoff depth (1961-1990)
tnrcc_ps	1839 point source dischargers within the Galveston Bay area (obtained from TNRCC)	Point	tnrcc_per_num	varies	TNRCC water quality permit number
			x-coord	varies	x location, in albers units
			y-coord	varies	y location in albers units
totshd	entire watershed area (delineated with burn), with San Jacinto Monument defined as the outlet	Poly.	none		
twcpr	TWC segmentation over the Upper HSC	Poly.	twc_id	1006	TNRCC water quality segment number
				1007	TNRCC water quality segment number

* NOTES:

The attributes are those in the pat or aat which are additional to the normal topology for those classes. And, all coverages are in USGS-Albers, unless otherwise specified

Grids *

Grid	Feature	Type	Z Units
acc_seg	Flow accumulation values along the HSC centerline	Floating Point	# of cells
ad	Grid of digitized USGS watershed for the Addicks Reservoir	Integer	n/a
bf	Grid of digitized USGS watershed for the Buffalo Bayou	Integer	n/a
bk	Grid of digitized USGS watershed for the Barker Reservoir	Integer	n/a
bodad	Areal BOD loading determined by this study for the Addicks Reservoir watershed	Floating Point	kg/yr/ha
bodann	Areal BOD loading over the entire Upper HSC watershed	Floating Point	kg/yr/ha
bodbf	Areal BOD loading determined by this study for Buffalo Bayou watershed	Floating Point	kg/yr/ha
bodbk	Areal BOD loading determined by this study for the Barker Reservoir watershed	Floating Point	kg/yr/ha
bodbr	Areal BOD loading determined by this study for the Brays Bayou watershed	Floating Point	kg/yr/ha
bodfac	Flow accumulation, weighted by BOD loading, over the Upper HSC watershed	Floating Point	kg/yr/ha
bodgr	Areal BOD loading determined by this study for the Greens Bayou watershed	Floating Point	kg/yr/ha
bodsc	Areal BOD loading determined by this study for the Ship Channel watershed	Floating Point	kg/yr/ha
bodsm	Areal BOD loading determined by this study for the Sims Bayou watershed	Floating Point	kg/yr/ha
bodwo	Areal BOD loading determined by this study for the Whiteoak Bayou watershed	Floating Point	kg/yr/ha
bod_out	BOD loading into each water quality segment	Integer	kg/yr/ha
bound1	Start and end points of the HSC study area	Integer	n/a
br	Grid of digitized USGS watershed for the Brays Bayou	Integer	n/a
cchint	Relative runoff coefficient value for station 8075900	Integer	unitless
chnnlfl	Flowlength values, just along the HSC study area	Floating Point	m
coeffac	Weighted flow accumulation of relative runoff coefficient over Upper HSC watershed	Floating Point	unitless
demfull	Geographic projection of 1:250,000 digital elevation model of four merged USGS DEMs: houston_west, seguin_east, beaumont_west, and austin_east	Floating Point	m
dlggrid	Grid of edited digital line graph used for HSC burn	Integer	n/a
emcbodgr	Grid of BOD EMC values for Upper HSC watershed	Floating Point	mg/L
emcbodin	Grid of BOD EMC values for Upper HSC watershed	Integer	mg/L
fcalc	Flow depth calculated from rainfall/flow/urbanization relation over Upper HSC ws.	Floating Point	mm/yr/ha
flfac	Flow accumulation, weighted by calculated flow, over the Upper HSC watershed	Floating Point	mm/yr/ha
flfacint	Flow accumulation, weighted by calculated flow, over the Upper HSC watershed	Integer	mm/yr/ha
flout	Total cumulative flow values at each water quality segment outlet	Floating Point	mm/yr/ha

Grid	Feature	Type	Z Units
f_p	Flow/Precipitation as calculated by the rainfall/flow/urbanization equation	Floating Point	unitless
gage	Grid of USGS gauge stations, moved onto 500 threshold streams	Integer	n/a
gage5900	Grid of gauge 8075900, moved onto 500 threshold streams	Integer	n/a
gage_shd	Subwatersheds for each USGS gauge station	Integer	n/a
gr	Grid of digitized USGS watershed for the Greens Bayou	Integer	n/a
grtotshd	Total watershed area with San Jacinto Monument defined as the outlet point, delineated from "burned" DEM	Integer	n/a
hscburn	DEM with the edited digital line graph "burned in"	Floating Point	m
hsccoef	Grid of relative runoff coefficients for Upper HSC watershed area	Floating Point	unitless
hscdem	Original DEM before stream "burn", Alber projection of demfull	Floating Point	m
hscdmalb	Albers projection of DEM, encompassing just beyond the watershed area	Floating Point	m
hscfac	Flow accumulation over the Upper HSC watershed, from hscfdr with stream "burn"	Floating Point	# of cells
hscfdr	Flow direction grid of the HSC watershed	Floating Point	n/a
hscfil	Filled DEM for the HSC watershed	Floating Point	m
hscfl	Flowlength of the HSC watershed	Floating Point	m
hscgrid1	Grid of just the centerline HSC, with all cells set to a value of 1	Integer	n/a
hsc_seg	HSC segmentation for study area	Integer	n/a
outlet	Grid of single outlet point of study area: San Jacinto Monument	Integer	n/a
out_seg	Outlet points defined for each channel segment	Integer	n/a
p1alb - p12alb	Average monthly precipitation grids (1961-1990) for each month	Integer	mm/month/ha
pannalb	Average annual precipitaion grid (1961-1990)	Integer	mm/yr/ha
pannchk	Cumulative annual precipitation value for gauge 8075900	Integer	mm/yr/ha
pannvin	Cumulative annual precipitation value for nine USGS gauges used in relationship	Integer	mm/yr/ha
rcalc	Calculated runoff from rainfall/runoff/precipitation equation over HSC watershed	Floating Point	mm/yr/ha
rfac	Flow accumulation, weighted by calculated runoff, over the Upper HSC watershed	Floating Point	mm/yr/ha
rstr	Average runoff depth over an area upstream of a given point on the delineated streams	Integer	mm/yr
r_annint	Calculated runoff from rainfall/runoff/precipitation equation over HSC watershed	Integer	mm/yr/ha
r_out	Total cumulative runoff values at each water quality segment outlet	Integer	mm/yr/ha
sc	Grid of digitized USGS watershed for the Ship Channel	Integer	n/a
seg_shd	Subwatersheds for each segment in the HSC study area	Integer	n/a

Grid	Feature	Type	Z Units
shd5900	Delineated watershed for gauge 8075900	Integer	n/a
sm	Grid of digitized USGS watershed for the Greens Bayou	Integer	n/a
str2000	Delineated streams, from burned DEM, at 2000 threshold	Integer	n/a
str500	Delineated streams, from burned DEM, at 500 threshold	Integer	n/a
strnobrn	Delineated streams from original DEM (w/o burn), 500 threshold	Integer	n/a
totshd5	Delineated watershed, from original DEM (no stream burn)	Integer	n/a
wo	Grid of digitized USGS watershed for the Whiteoak Bayou	Integer	n/a

* NOTES:
All grids are 100mx100m cells, in USGS-Albers, unless otherwise specified

Shape Files

Shape File	Feature	Class	Coverage
hydroseg.shp	Hydrographic segmentation over the Upper HSC (Ward and Armstrong, 1992)	Poly.	n/a
bc_seg.shp	Boundary water quality modeling segments for Upper HSC	Arc	bcseg
ps_msrd.shp	68 point source dischargers along the Upper HSC	Point	hsc_pscv

Tables

Table	Feature	Alias*	Attribute	Description	Units
bnd_seg.dbf	dBase File which contains boundary segment modeling parameters	Boundary Segments	grid-code	Boundary segment #	n/a
			ex_coeff	Exchange coefficient	m^2/sec
			x_area	Cross-sectional area	m^2
			upstr_seg	Upstream segment	n/a
			dwnstr_seg	Downstream segment	n/a
			bttm_seg	Segment below this segment	n/a
			int_bod	BOD boundary concentration	mg/L
			int_do	DO boundary concentration	mg/L
			sod	Sediment oxygen demand	g/m^2-day
			temp	Temperature	° C
			sal	Salinity	ppt
			act_length	Length of boundary segment	m
			depth	Depth of segment	m
			type	Type of segment (WASP5 standards)	n/a
			perpin	Y or N to indicate orientation to main channel	n/a
			name	Descriptive name of the boundary segment	n/a
bc_seg.dbf	dBase File attached to bc_seg shape file	n/a	No additional attributes		
bod_out.vat	vat of BOD flow accumulation value for each segment outlet	BOD Loading Values	value	Arc assigned cell value	n/a
			count	# of cells with this value	n/a
			grid-code	Main segment #	n/a
			bodfac	accumulated bod loading	kg/yr/ha
bod_ps.dbf	dBase File of total point source loading into each segment	Point Source BOD	segment	Main segment #	
			BOD	BOD loading	10^3 lbs/yr
bodavyr.dbf	dBase File of modeling results of BOD for average year conditions	n/a	Time	Time	days
			1 through 17	BOD concentration for the respective segment #	mg/L
boddry.dbf	dBase File of modeling results of BOD for dry weather conditions	n/a	Time	Time	days
			1 through 17	BOD concentration for the respective segment #	mg/L
bodhgk2.dbf	dBase File of modeling results of BOD for average year conditions, high k_2	n/a	Time	Time	days
			1 through 17	BOD concentration for the respective segment #	mg/L
bodhgkd.dbf	dBase File of modeling results of BOD for average year conditions, high k_d	n/a	Time	Time	days
			1 through 17	BOD concentration for the respective segment #	mg/L
doavyr.dbf	dBase File of modeling results of DO	n/a	Time	Time	days

Table	Feature	Alias*	Attribute	Description	Units
	for average year conditions		1 through 17	DO concentration for the respective segment #	mg/L
dodry.dbf	dBase File of modeling results of DO	n/a	Time	Time	days
	for dry year conditions		1 through 17	DO concentration for the respective segment #	mg/L
dohgk2.dbf	dBase File of modeling results of DO	n/a	Time	Time	days
	for average year conditions, high k_2		1 through 17	DO concentration for the respective segment #	mg/L
dohgkd.dbf	dBase File of modeling results of DO	n/a	Time	Time	days
	for average year conditions, high k_d		1 through 17	DO concentration for the respective segment #	mg/L
flout.vat	vat of flow, flow accumulation value	Flow	value	Arc assigned cell value	n/a
	for each segment outlet	Accumulation	count	# of cells with this value	n/a
		Values	grid-code	Main segment #	n/a
			flfac	accumulated flow depth	mm/yr/ha
flow.dbf	dBase File created during Input Block D	n/a	grid-code	Main segment #	n/a
	execution in ArcView connection		cumm_flow	Cumulative calculated flow	m^3/sec
			int_flow	Incremental calculated flow	m^3/sec
			runoff	Incremental calculated runoff	m^3/sec
			baseflow	Incremental calculated baseflow	m^3/sec
gbayps.dat	INFO file of dischargers into Galveston Bay	n/a	Table not available for publication		
hsc_dis.dbf	dBase File of dischargers into Galveston Bay	n/a	Table not available for publication		
hsc_res.dbf	dBase File summarizing modeling	n/a	grid-code	Main segment #	n/a
	results for Upper HSC		do_aveyr	Modeled DO for average year conditions	mg/L
			bod_aveyr	Modeled BOD for average year conditions	mg/L
			do_drywtr	Modeled DO for dry weather conditions	mg/L
			bod_drywtr	Modeled BOD for dry weather conditions	mg/L
			salinity	Modeled salinity	ppt
hsc_sal.dbf	dBase File of modeling results of calibration	n/a	Time	Time	days
			1 through 17	Salinity concentration for the respective segment #	ppt
hyd_seg.dbf	dBase File summarizing measured values		segment_id	Hydrographic segment #	n/a
	for hydrographic segmentation		DO	Average measured DO	mg/L
	(Ward and Armstrong, 1992)		BOD	Average measured BOD	mg/L
			salinity	Average measured salinity	ppt
hydroseg.dbf	dBase File attached to hydroseg shape file	n/a	No additional attributes		
load.dbf	dBase File created during Input Block F	n/a	grid-code	Main segment #	n/a
	execution in ArcView connection		bod_ps	Total point source loading into main segment	kg/day

Table	**Feature**	**Alias***	**Attribute**	**Description**	**Units**
main_seg.dbf	dBase File which contains main segment modeling parameters	Main Segment Parameters	grid-code	Main segment #	n/a
			ex_coeff	Exchange coefficient	m^2/sec
			x_area	Cross-sectional area	m^2
			upstr_seg	Upstream segment	n/a
			dwnstr_seg	Downstream segment	n/a
			bttm_seg	Segment below this segment	n/a
			int_bod	BOD initial concentration	mg/L
			int_do	DO initial concentration	mg/L
			sod	Sediment oxygen demand	g/m^2-day
			temp	Temperature	° C
			sal	Salinity	ppt
			width	With of main segment	m
			depth	Depth of segment	m
			type	Type of segment (WASP5 standards)	n/a
ps_msrd.dbf	dBase File of the dischargers along the HSC with measurements -- attached to shape file	Same as uphscps.dbf	Table not available for publication		
r_out.vat	vat of runoff flow accumulation value for each segment outlet	Runoff Accumulation Values	value	Arc assigned cell value	n/a
			count	# of cells with this value	n/a
			grid-code	Main segment #	n/a
			rfac	accumulated runoff depth	mm/yr/ha
uphscps.dbf	dBase File of the dischargers along the HSC with measurments	Same as ps_msrd.dbf	Table not available for publication		

* Alias refers to name needed for ArcView/WASP5 connection

Appendix B
Projection File: geoalb.prj

```
input
projection geographic
units dd
parameters
output
projection albers
units meters
spheroid grs80
datum NAD83
parameters
29 30 00
45 30 00
-96 00 00
23 00 00
0.0
0.0
end
```

Appendix C
Programs, Macros, And Procedures

Appendix C-1
Procedure and Macro for Importing Digital Line Graphs from CD ROM to ArcInfo

Source: Saunders, 1996

```
/*  An ARC AML FOR PREPARING DLG DATA FOR REGIONAL ANALYSIS
/*  AML NAME:  dlghydro.aml (run from the "Arc" prompt)
/*  FUNCTION:  Prepares selected DLG data for analysis with respect to a
/*                  particular hydrologic or political region.
/*  INPUTS:
/*        -all compressed ("zipped") DLG files corresponding to the region of
/*           interest.  These zipped files are downloaded from the USGS EROS
/*           Data Center at http://sun1.cr.usgs.gov/eros-home.html.
/*           Alternatively the DLG files can be accessed from US Geodata
/*           1:100,000-Scale DLG Data Compact Disc (USGS, 1993).
/*        -a projection file that will allow for conversion from utm map
/*           coordinates to whatever projection is desired.
/*        -a polygon coverage delineating the boundary of the hydrologic or
/*           political region of interest.
/*  AUTHOR:
/*        William Saunders, Graduate Student, University of Texas at Austin,
/*        Environmental and Water Resources Division, Department of Civil
/*        Engineering, April 1996
/*  MODIFIED BY:
/*        Jennifer Benaman, Graduate Student, University of Texas at Austin,
/*        Enviromental and Water Resources Division, Department of Civil
/*        Engineering, April 1996
/********************************************************************************
/*  BEGIN AML EXECUTION
/*
/*  Assuming that 4 zipped DLG hydro files have been downloaded (in this case
/*  from CD-ROM using the following commands:
/*
/*  cp /cdrom/100k_dlg/beaumont/bm4hydro.zip ./
/*  cp /cdrom/100k_dlg/conroe/bm3hydro.zip ./
/*  cp /cdrom/100k_dlg/anahuac/ho2hydro.zip ./
/*  cp /cdrom/100k_dlg/houston/ho1hydro.zip ./
/*
/*  This first set of commands are the only one that the user of the AML must
/*  change.  Store the number of zipped DLG files into the variable dlgnum.
/*  Then, for each zipped DLG file, define sequential variables called dlg# as
/*  the first 3 characters of each of the zipped files.
/*
&sv dlgnum = 4
&sv dlg1 = ho1
&sv dlg2 = ho2
&sv dlg3 = bm3
&sv dlg4 = bm4
/*
/*  This part of the AML unzips all of the compressed files to create 15-minute
/*  map files.  Each 15-minute map file is first converted into an Arc/Info
/*  line coverage.  Then, the borders of each of the 15-minute map files are
/*  trimmed away from the coverage so that those 15-minute meridians and
/*  parallels will not appear in the final appended coverage.
/*
&sv count = 1
&do &while %count% le %dlgnum%
  &sv filename = [value dlg%count%]
  &sv count = %count% + 1
  &sys unzip %filename%hydro.zip
  &sv count2 = 1
  &do &while %count2% le 8
     &do &while [exists %filename%hyf0%count2% -file]
       dlgarc optional %filename%hyf0%count2% %filename%f0%count2%
       &sv x = [delete %filename%hyf0%count2% -file]
       build %filename%f0%count2% line
       reselect %filename%f0%count2% %filename%0%count2% line # line
       res rpoly# > 1
       ~
       n
       y
       res lpoly# > 1
       ~
       n
       n
       kill %filename%f0%count2% all
     &end
     &sv count2 = %count2% + 1
  &end
&end
/*
/*  This part of the AML merges, or "appends", all of the 15-minute map file
/*  coverages together and then builds line topology for the resultant
/*  coverage,
/*  called "bigmap".
/*
append bigmap
&sv count = 1
&do &while %count% le %dlgnum%
  &sv filename = [value dlg%count%]
  &sv count = %count% + 1
  &sv count2 = 1
  &do &while %count2% le 8
     &do &while [exists %filename%0%count2% -cover]
       %filename%0%count2%
       &sv count2 = %count2% + 1
     &end
  &sv count2 = %count2% + 1
  &end
&end
~
y
y
build bigmap line
/*
/*  Once "bigmap" has been created, each of the coverages that were merged to/*
build it are no longer necessary.  This part of the AML kills off all of
/*  the intermediate level coverages used to append "bigmap".
/*
&sv count = 1
&do &while %count% le %dlgnum%
  &sv filename = [value dlg%count%]
  &sv count = %count% + 1
  &sv count2 = 1
  &do &while %count2% le 8
     &do &while [exists %filename%0%count2% -cover]
       kill %filename%0%count2% all
       &sv count2 = %count2% + 1
     &end
  &sv count2 = %count2% + 1
  &end
&end
/*
/*  The "bigmap" coverage is then reprojected to the desired map projection
/*  and coordinates.  The example below shows projection from Universal
/*  Transverse Mercator coordinates (the projection of the files as they exist
/*  on CDROM or from the Internet site) to the Texas State Mapping System.
/*  Accordingly, this projection file must be located in the path specified
/*  in the project statement.  For this statement, the utmtsms.prj file is
/*  located in the same directory as the "bigmap" coverage.
/*
project cover bigmap hydrocov utmtsms.prj
/*
/*  Finally, a polgyon coverage of the hydrologic or political boundary of
/*  interest is used to "clip" out the hydrologic features specific to that
/*  region.
/*
clip hydrocov border reghydro line
/*
&return
/*******************************end of AML**************************************
```

Appendix C-2
Procedure to Obtain USGS Land Use/Land Cover Files from USEPA Ftp Site

Source: Maidment, 1996

Introduction

Land Use/Land Cover (LULC) files are developed by the USGS and are available via internet from either USGS or the EPA. In both cases, the user must specify the 1:250,000-scale mapsheet name(s) corresponding to the region(s) of interest, and then download the coverage(s). Accessing the actual LULC files from the USGS has proven problematic to date, as the files exist in a "modified" UTM projection that is not clearly defined in the users guide. Given these difficulties and the relative ease with which the same files are obtained from EPA, the EPA Home Page is currently the preferred source of this data. However, the 1:250,000-scale mapsheet names for the United States are still accessible at the USGS Home Page.

The following procedure is a THREE step process and may not sound very efficient. However, it is the only way we have found, so far, to successfully obtain the LULC files.

STEP 1 Accessing the 1:250,000-scale mapsheet names:

- Once at the initial Home Page of a particular data browser (Mosaic, Netscape, etc.), enter the address of the USGS EROS Data Center Home Page - http://sun1.cr.usgs.gov/eros-home.html.

- At the Home Page, scroll down to the 1:250,000-Scale and 1:100,000-Scale LULC section of the page and select 250K FTP via Graphics. A map of the continental United States will appear.

- Using the cursor, click in the general area of the particular region of interest, and a finer resolution map will appear with the 1:250,000-scale maps and mapnames superimposed.

- Note the mapnames that correspond to the hydrologic or political region of interest.

STEP 2 Obtaining the Filenames from EPA:

- An alternate source for LULC data is the EPA Home Page - http://www.epa.gov.

Retrieval of LULC files from this location still requires mapsheet names to be known ahead of time (see step 1 above).

- Once at this Home Page, select the Search the EPA Server hypertext. When you click on this hypertext, a query box will appear; within the query box, type *land use* and then press 'start query'. The result of the query should appear after a minute or two.

- Scroll down to and select the EPA EPAGIRAS hypertext and then enter a mapsheet name of a LULC coverage to be downloaded. This will lead to a page where an export file (.e00.gz) should be available for selection.

- write down the actual name of the e00 file -- it should be a series of three letters and 5 numbers, followed by .e00.gz.

STEP 3 Obtaining the file from the EPA ftp site

Since downloading the file through Netscape or Mosaic is not always successful (especially in the middle of the day). It is best to go to the EPA ftp site. Now that you know the actual filename of what you are looking for, this process is pretty straightforward.

- At a command prompt, type ftp earth1.epa.gov

- Logon anonymously by typing anonymous at the user name prompt and your e mail address for the password

- Change directories to /pub/EPAGIRAS/mgiras (note that ftp is case sensitive!)

- Once in this directory, you can type get filename. The download process will take some time. Typically, these files are between 1.5 and 2 megs. When you are done getting the files you need, exit out of the ftp prompt by typing bye. - At the Unix prompt of the host workstation, unzip the compressed file:

$: **gunzip filename.e00.gz**

Invoke Arc/Info and import the file as an Arc/Info coverage:

Arc: **import cover filename.e00 filename**

The process is now complete as the LULC coverage exists in an Albers projection that can easily be manipulated with other coverages and LULC files.

Appendix C-3
FORTRAN Program Used to Estimate Baseflow from Measured Flow

Source: Olivera, *pers. comm.*, 1996

```
***********************************************************************
*   Author:  Francisco Olivera
*          CRWR -- University of Texas at Austin
*   Date:    February 5, 1996
*   Revised: Jennifer Benaman
*          CRWR -- University of Texas at Austin
*   Date:    February 16, 1996
*   Purpose: Given a flow time series, the program generates
*          base flow and direct runoff time series.
*          Given the plot of the flwo time series, the base flow is
*          obtained by pivoting a straight line ona point and
*          connecting it with the lowest part of the flow curve.  The
*          length of the straight line is defined by the user.
*
*   Input:   (1)  Length of the straight line (in time steps)
*               (2)  Flow time series
************************************************************************
          PROGRAM BFLOW3

*  Declaring Parameters

          PARAMETER (NMAX=10000)

*  Declaring variables

          REAL FLOW(NMAX), BASEFLOW(NMAX), MINSLOPE, SLOPE(NMAX)
          INTEGER L,N

*  Open input files for reading

          OPEN(UNIT=10, FILE='FLOW.IN', STATUS='OLD')

*  Open output file for writing

          OPEN(UNIT=30, FILE='BASEFLOW.OUT', STATUS='UNKNOWN')

*  Reading input file

          READ(10,*) L
          DO 101 I=1,NMAX
            READ(10,*,END=901) FLOW(I)
            N=I
 101  CONTINUE
 901  CONTINUE

*  Generating the baseflow

          BASEFLOW(1)=FLOW(1)
          I=1
 501  CONTINUE
          MINSLOPE=1000000

          DO 102 J=1,L
            IF (I+J .LE. N) THEN
                    SLOPE(J) = (FLOW(I+J)-BASEFLOW(I))/J
                    IF (SLOPE(J) .LT. MINSLOPE) THEN
                      MINSLOPE=SLOPE(J)
                    END IF
            END IF
 102  CONTINUE

          BASEFLOW(I+1)=BASEFLOW(I)+MINSLOPE

          I=I+1
          IF(I-N) 501,502,502
 502  CONTINUE

*   Echo the output to the screen

          DO 104 I=1,N
                    WRITE (*,*) I,FLOW(I),BASEFLOW(I)
 104  CONTINUE

*   Writing the Flow and the Baseflow

          DO 105 I=1,N
                    WRITE(30,*) I, FLOW(I), BASEFLOW(I)
 105  CONTINUE

          END
```

Appendix C-4
FORTRAN Programs Used to Format Text Files for WASP5 Input File

Program: calgen.for
Generates calibration input file for TOXI5

```
*******************************************************************
*      Program:  Used to take ten text files written from
*                ArcView and write the WASP5 input file.
*                Presently, this program only writes the file
*                for TOXI5 -- model calibration w/salinity.
*      Input:    Twelve text files
*      Author:   Jennifer Benaman
*                Graduate Research Assistant
*      Date:     June, 1996
*******************************************************************

* Main Program
      PROGRAM INPUTCAL

* Declare Variables

      REAL A(20), B(100,5), C(100,4),D(300,2),E(100,2),F(100,2),
     &  G(100,4),H(10),I(10),J(300,2),TPRINT
      INTEGER NOSEG,MNSEG
      CHARACTER*12 FNAME

* Explanation of Variables

*     A(20): the array which holds the input file A
*                 variables: model paramters
*     B(100,5):  the array which holds the input file
*                 B variables: exchange functions
*     C(100,4):  the array which holds the input file
*                 C variables: volumes
*     D(300,2):  the array which holds the input file
*                 D variables: flows
*     E(100,2):  the array which holds the input file
*                 E variables: boundaries
*     F(100,4):  the array which holds the input file
*                 F variables: loads
*     G(100,4):  the array which holds the input file
*                 G variables: parameters
*     H(10):     the array which holds the input file
*                 H variables: constants
*     I(10):     the array which holds the input file
*                 I variables: times functions
*     J(300,2):  the array which holds the input file
*                 G variables: initial conditions
*     TPRINT: the number of days the model is to be
*                 run -- obtained in subroutine INPUTA
*     NOSEG:  the number of segments in the system -- obtained
*                 in Subroutine INPUTA

* Open Files

      OPEN (UNIT=6, FILE='INPTNME.TXT',STATUS='OLD')
      READ (6,16) FNAME
   16 FORMAT (A12)

      OPEN (UNIT=3, FILE='TITLE.TXT',STATUS='OLD')
      OPEN (UNIT=5, FILE='CALA.TXT',STATUS='OLD')
      OPEN (UNIT=10, FILE='B.TXT',STATUS='OLD')
      OPEN (UNIT=15, FILE='C.TXT',STATUS='OLD')
      OPEN (UNIT=13, FILE='DYN.TXT',STATUS='OLD')
      OPEN (UNIT=20, FILE='D.TXT',STATUS='OLD')
      OPEN (UNIT=25, FILE='CALE.TXT',STATUS='OLD')
      OPEN (UNIT=30, FILE='CALF.TXT',STATUS='OLD')
      OPEN (UNIT=35, FILE='CALG.TXT',STATUS='OLD')
      OPEN (UNIT=40, FILE='CALH.TXT',STATUS='OLD')
      OPEN (UNIT=45, FILE='I.TXT',STATUS='OLD')
      OPEN (UNIT=50, FILE='CALJ.TXT',STATUS='OLD')
      OPEN (UNIT=55, FILE=FNAME,STATUS='UNKNOWN')

* Call Subroutines

      CALL INPUTA(A,TPRINT,NOSEG,MNSEG)
      CALL INPUTB(B,TPRINT,NOSEG)
      CALL INPUTC(C,NOSEG)
      CALL INPUTD(D,TPRINT)
      CALL INPUTE(E,TPRINT)
      CALL INPUTF(F,TPRINT,MNSEG)
      CALL INPUTG(G,NOSEG)
      CALL INPUTH(H)
      CALL INPUTI(I)
      CALL INPUTJ(J,NOSEG)

      STOP
      END

******************** Subroutines
```

```
***Subroutine INPUTA:  Creates the A Block to the WASP5 input file
***The following defaults are set in this subroutine:
******Backward differencing is always used (ADFAC=0.0)
******A transport file is always generated (TFLG = 0)
******The first six segments are those which solns are displayed
*     on the screen (Record 5)
******The same maximum time step is used throughout the model-run
******The same print interval is used throughout the model-run
***This subroutine also reads the time the model will run (TPRINT)

      SUBROUTINE INPUTA(A,TPRINT,NOSEG,MNSEG)

      INTEGER NOSEG, NOSYS, ICFL, MFLAG, JMASS, NEGSLN, INTYP,
     & TFLG,ZMIN, ZHR,
     & MNSEG
      REAL A(11), ADFAC, ZDAY, DTS, TPRINT, PRTINV
      CHARACTER*80 TITLE
      CHARACTER*75 LEVEL

      READ (3,1) TITLE
      READ (5,*) A

* Declaring Variables


      NOSEG = INT(A(1))
      NOSYS = 1
      ICFL = INT(A(2))
      MFLAG = INT(A(3))
      JMASS = 1
      NEGSLN = INT(A(4))
      ZDAY = A(5)
      ZHR = INT(A(6))
      ZMIN = INT(A(7))
      INTYP = 1
      TFLG = 0
      ADFAC = 0.0
      DTS = A(8)
      PRTINV = A(10)
      TPRINT = A(9)
      MNSEG = INT(A(11))
      LEVEL = 'Calibration with Salinity'

* Format statements

    1    FORMAT (A80)
    2    FORMAT (A5,1X,A)
    3    FORMAT (11A5,A25)
    4    FORMAT (7I5,2F5.0,I3,I2,I5)
    5    FORMAT (8I5)
    6    FORMAT (I5)
    7    FORMAT (2(F10.1, F10.1))
    9    FORMAT (F10.5,F10.1)
```

```
* Writing the input file

      WRITE (55,1) TITLE
      WRITE (55,2) "TOXI5 ",LEVEL
      WRITE (55,3) "NSEG","NSYS","ICRD","MFLG","IDMP","NSLN",
     & "INTY","ADFC","DD","HHMM","TFLG","A: MODEL OPTIONS"
      WRITE (55,4) NOSEG, NOSYS, ICFL, MFLAG, JMASS, NEGSLN,
     & INTYP, ADFAC, ZDAY, ZHR, ZMIN, TFLG
      WRITE (55,5) 1,2,3,4,5,6
      WRITE (55,6) 1
      WRITE (55,9) DTS,TPRINT
      WRITE (55,6) 2
      WRITE (55,7) PRTINV,0.,PRTINV,TPRINT
      WRITE (55,5) 0,1,1,1,1,1,1

      RETURN
      END

***Subroutine INPUTB:  Creates the B Block to the WASP5 input file
***The following defaults are set in this subroutine:
******All exchanges are steady state
******The scaling and conversion factors are set to 1.0 (Areas should
*     be in m^2 and exchange coeff. in m^2/sec)

      SUBROUTINE INPUTB(B,TPRINT,NOSEG)

      INTEGER NRFLD,NTEX,NORS,
     + TEMP,S, N
      REAL B(100,5),TPRINT

      COUNT = 1
      DO WHILE (TEMP2 .NE. 555)
         READ (10,*) B(COUNT,1)
         TEMP2 = B(COUNT,1)
         COUNT = COUNT + 1
      END DO

      REWIND (UNIT = 10)
      N = COUNT-2
      DO I=1,N
              READ (10,*) (B(I,J),J=1,5)
      END DO

      NRFLD = INT(B(1,2))

      WRITE (55,10) NRFLD,"B:EXCHANGES"

   10 FORMAT (I5,5X,A)
   11 FORMAT (I5,2F10.1)
   12 FORMAT (I5)
   13 FORMAT (2F10.1,2I5)
   14 FORMAT (4(F10.1,F10.1))
   15 FORMAT (8I5)
```

```
      NTEX = 1
      TEMP = 2
      DO I = TEMP, N-1
          IF (B(I,1).EQ.B(I+1,1)) THEN
          GO TO 500
          ELSE
          NTEX = NTEX + 1
          END IF
  500 END DO

      WRITE (55,11) NTEX,1.0,1.0

      DO S = 1,NTEX

        NORS = 1
          DO I = TEMP,N-1
              IF (B(I,1) .EQ. B(I+1,1)) THEN
              NORS = NORS + 1
              ELSE
              GO TO 510
              END IF
          END DO

  510   WRITE (55,12) NORS

          DO I = TEMP,TEMP+NORS-1
              WRITE (55,13) B(I,2),B(I,3),INT(B(I,4)),INT(B(I,5))
          END DO
        WRITE (55,12) 2
        WRITE (55,14) B(TEMP,1),0.0,B(TEMP,1),TPRINT
        TEMP = TEMP + NORS

      END DO

      WRITE (55,15) 0,1,1,1,1,1,1,1
      RETURN
      END

***Subroutine INPUTC:  Creates the C Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1.0 (Volumes should
*     be in m^3)
******The hydraulic coefficients used to calculate reaeration and
*     volitilization do not spatially vary

      SUBROUTINE INPUTC(C,NOSEG)

      INTEGER NOSEG, N
      REAL C((NOSEG+5),4)

      N = NOSEG + 5

      DO I=1,N
              READ (15,*) (C(I,J),J=1,4)
      END DO

   20 FORMAT (2I5,F10.4,5X,A)
   21 FORMAT (2F10.1)
   22 FORMAT (3I10,F10.1,F10.4,3F10.1)

      WRITE (55,20) INT(C(1,1)),INT(C(1,2)),C(1,3),"C:VOLUMES"
      WRITE (55,21) 1.0,1.0

      DO I = 6,N
          WRITE (55,22) INT(C(I,1)),INT(C(I,2)),INT(C(I,3)),C(I,4),
     &                  C(2,1),C(3,1),C(4,1),C(5,1)
      END DO

      RETURN
      END

***Subroutine INPUTD:  Creates the D Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1.0 (Flows should
*     be in m^3/sec)
******The number of flow fields is set to 1:  Water column only (no pore
*     water flows
******The flow is steady state


      SUBROUTINE INPUTD(D,TPRINT)

      INTEGER  I,IQOPT,NFIELD,K,J,
     & W,COUNT,X,FLW,H
      REAL D(300,2),TPRINT,TEMP
      CHARACTER*12 HYDFILE

      NFIELD = 1
      FLW = 0
      N = 2
      TEMP = 0.0
      DO WHILE (TEMP.NE.555.0)
          READ (20,*) (D(N,I),I=1,2)
          IF (D(N,2).EQ.999.0) THEN
              FLW = FLW+1
              PRINT *, FLW
          END IF
          TEMP = D(N,1)
          N = N+1
      END DO
      REWIND (UNIT = 20)

      DO I = 1,1
           READ(20,*) (D(I,J),J=1,2)
      END DO

      READ (13,35) HYDFILE
```

```
    IQOPT = INT(D(1,1))


    IF (IQOPT .EQ. 3) THEN
        WRITE (55,30) IQOPT,NFIELD,HYDFILE,"D:FLOWS"
        RETURN
    ELSE
        WRITE (55,30) IQOPT,NFIELD,HYDFILE,"D:FLOWS"
        WRITE (55,31) FLW, 1.0, 1.0
    END IF

    Z = 1
    X = 2
    TEMP = 0.0
    W = 1
    DO WHILE (TEMP.NE.555.0)
    COUNT = 0
    DO I = X,300
        READ(20,*) (D(I,J),J=1,2)
        IF (D(I,2).EQ.999) THEN
            GO TO 800
        ELSE IF (D(I,1).EQ.555.0) THEN
            GO TO 810
        ELSE
            COUNT = COUNT + 1
            W = W+1
        END IF
    END DO


800 WRITE (55,32) COUNT

    DUMMY1 = INT(COUNT/4)
    DUMMY2 = COUNT - DUMMY1*4

    IF (DUMMY1.EQ.0) THEN
        DUMMY3 = X
    END IF

    DO K = X,(4*DUMMY1)+X-1,4
        WRITE (55,33) 1.0,INT(D(K,1)),INT(D(K,2)),
   &     1.0,INT(D(K+1,1)),INT(D(K+1,2)),
   &     1.0,INT(D(K+2,1)),INT(D(K+2,2)),
   &     1.0,INT(D(K+3,1)),INT(D(K+3,2))
         DUMMY3 = K+4
    END DO

    IF (DUMMY2.NE.0) THEN
    WRITE (55,33) (1.0,INT(D(H,1)),INT(D(H,2)),H=DUMMY3,DUMMY2+
   & DUMMY3-1)
    END IF

    WRITE (55,32) 2
    WRITE (55,34) D(W+Z,1),0.0,D(W+Z,1),TPRINT
```

```
      X = COUNT+X+1
      Z = Z+1
  810 TEMP = D(I,1)
      END DO

   30 FORMAT (2I5,A12,5X,A)
   31 FORMAT (I5,2F10.1)
   32 FORMAT (I5)
   33 FORMAT (4(F10.2,2I5))
   34 FORMAT (4F10.2)
   35 FORMAT (A12)
   36 FORMAT (8I5)

      WRITE (55,36) 0,1,1,1,1,1,1,1

      RETURN
      END

***Subroutine INPUTE:  Creates the E Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1000 (Boundary conditions
*     should be in ppt)
******Only Salinity is considered
******The bc's are steady state


      SUBROUTINE INPUTE(E,TPRINT)

      INTEGER  NOBC,I,
     & N
      REAL E(100,2),TPRINT

      DO I = 1,1
           READ(25,*) (E(I,J),J=1,2)
      END DO

      NOBC = INT(E(1,1))
      N = NOBC + 1

      IF (NOBC .EQ. 0) THEN
          WRITE (55,40) 0,"*","E:BOUNDARIES"
          RETURN
      END IF

      DO I = 2,N
          READ(25,*) (E(I,J),J=1,2)
      END DO

   40 FORMAT (I10,2X,A5,5X,A)
   42 FORMAT (2I5)
   43 FORMAT (2(2F10.2))
```

```
      WRITE (55,40) NOBC,"SAL","E:BOUNDARIES"
      WRITE (55,43) 1000.0,1.0
      DO I = 2,N
          WRITE (55,42) INT(E(I,1)),2
          WRITE (55,43) E(I,2),0.0,E(I,2),TPRINT
      END DO


      RETURN
      END

***Subroutine INPUTF:  Creates the F Block to the WASP5 input file
***The following defaults are set in this subroutine:
******There are no point or nonpoint source salinity loadings


      SUBROUTINE INPUTF(F,TPRINT,MNSEG)

      INTEGER  N,
     & MNSEG
      REAL F(25,2),TPRINT

      READ (30,*) F(1,1)


      N = INT(F(1,1))

      IF (N .EQ. 0) THEN
          WRITE (55,60) N,"F:LOADS -- NO LOADS"
          WRITE (55,60) 0,"NO NPS LOADS"
          RETURN
      END IF

   60 FORMAT (I10,5X,A)

      RETURN
      END

***Subroutine INPUTG:  Creates the G Block to the WASP5 input file
***The following defaults are set in this subroutine:
******There are no paramters for TOXI5 complexity level 1

      SUBROUTINE INPUTG(G,NOSEG)

      INTEGER NOSEG,NOPAR
      REAL G((NOSEG+1),4)

      READ(35,*) G(1,1)


      NOPAR = INT(G(1,1))

   50 FORMAT (I10,5X,A)
```

```
      WRITE (55,50) NOPAR,"G:PARAMETERS -- NO PARAMETERS"

      RETURN
      END

***Subroutine INPUTH:  Creates the H Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The constants do not spatially vary
******Partition coefficient in L/kg
******Water column biodegradation
******Molecular Weight in g/mole


      SUBROUTINE INPUTH(H)

      REAL H(3),PIXC,KBW,MOLWT

      READ(40,*) H

      PIXC = H(1)
      KBW = H(2)
      MOLWT = H(3)

   70 FORMAT (10X,A)
   71 FORMAT (A10,I10)
   72 FORMAT (2(A10,I10,F10.2))
   73 FORMAT (A10,I10,F10.2)

      WRITE (55,70) "H:CONSTANTS"
      WRITE (55,71) "GLOBALS",0
      WRITE (55,71) "SALINITY",1
      WRITE (55,72) "GENERAL",3
      WRITE (55,72) "PIXC",111,PIXC,"KBW",141,KBW
      WRITE (55,73) "MOLWT",81,MOLWT

      RETURN
      END

***Subroutine INPUTI:  Creates the I Block to the WASP5 input file
***The following defaults are set in this subroutine:
******Since this model is considered steady state (right now), there
*     are no time functions presently input into the model


      SUBROUTINE INPUTI(I)

      REAL I(1)

      READ (45,*) I

      WRITE (55,80) INT(I(1)),"I:TIME FUNCTIONS"
   80 FORMAT (I10,20X,A)

      RETURN
      END

***Subroutine INPUTJ:  Creates the J Block to the WASP5 input file
***The following defaults are set in this subroutine:
******This program sets all i.c.'s to 0, except for those boundary
*     conditions set in ArcView
******The dissolved fraction of salinity is 1.0
******The maximum value of all variables is set at 35000 mg/L
******The density of salinity is set at 0.0 -- EUTRO does not
*     use those numbers
******All initial conditions are in mg/L


      SUBROUTINE INPUTJ(J,NOSEG)

      INTEGER  NOSEG,K,NOBC,N,L
      REAL J(NOSEG*8+8,2),DENBOD,DENDO,MAX,
     & FRAC,TEMP(30,2)

      DO K = 1,2
          READ (50,*) (J(K,L),L=1,2)
      END DO

      FRAC = J(1,1)
      NOBC = INT(J(2,1))
      DO K = 3,2+NOBC
          READ (50,*) (J(K,L),L=1,2)
      END DO

      N = 3
      DO K = 1,NOSEG
              IF (K.EQ.INT(J(N,1))) THEN
              TEMP(K,1) = J(N,1)
              TEMP(K,2) = J(N,2)*1000
              N = N+1
              ELSE
              TEMP(K,1) = REAL(K)
              TEMP(K,2) = 0.0
              END IF
      END DO


      MAX = 35000.0
      DENBOD = 1.0
      DENDO = 1.0

   90 FORMAT (A10,30X,I5,F5.1,E10.2,A)
   91 FORMAT (3(I5,2F10.2))

      DUMMY1 = INT(NOSEG/3)
      DUMMY2= NOSEG - 3*DUMMY1

      WRITE (55,90) "SAL",3,0.0,MAX,"J:INITIAL CONDITIONS"
      DO K = 1,3*DUMMY1,3
          WRITE (55,91) INT(TEMP(K,1)),TEMP(K,2),FRAC,
     &         INT(TEMP(K+1,1)),TEMP(K+1,2),FRAC,
     &         INT(TEMP(K+2,1)),TEMP(K+1,2),FRAC
      DUMMY3 = K+3
      END DO
      WRITE (55,91) (INT(TEMP(K,1)),TEMP(K,2),FRAC,K=DUMMY3,DUMMY2
     &    +DUMMY3-1)


      RETURN
      END
```

Program: outgen.for
Generates BOD/DO model input file for EUTRO5

```
******************************************************************
*      Program:  Used to take ten text files written from
*                ArcView and write the WASP5 input file.
*                Presently, this program only writes the file
*                for EUTRO5.
*      Input:    Ten text files
*      Author:   Jennifer Benaman
*                Graduate Research Assistant
*      Date:     June, 1996
******************************************************************
*
* Main Program
      PROGRAM INPUTWASP
```

```
* Declare Variables

      REAL A(20), B(100,5), C(100,4),D(300,2),E(100,2),F(100,2),
     &  G(100,4),H(10),I(10),J(300,2),TPRINT
      INTEGER NOSEG,MNSEG
      CHARACTER*12 FNAME

* Explanation of Variables

*     A(20): the array which holds the input file A
*                 variables: model paramters
*     B(100,5):  the array which holds the input file
*                 B variables: exchange functions
*     C(100,4):  the array which holds the input file
*                 C variables: volumes
*     D(300,2):  the array which holds the input file
*                 D variables: flows
*     E(100,2):  the array which holds the input file
*                 E variables: boundaries
*     F(100,4):  the array which holds the input file
*                 F variables: loads
*     G(100,4):  the array which holds the input file
*                 G variables: parameters
*     H(10):     the array which holds the input file
*                 H variables: constants
*     I(10):     the array which holds the input file
*                 I variables: times functions
*     J(300,2):  the array which holds the input file
*                 G variables: initial conditions
*     TPRINT: the number of days the model is to be
*                 run -- obtained in subroutine INPUTA
*     NOSEG:  the number of segments in the system -- obtained
*                 in Subroutine INPUTA

* Open Files

      OPEN (UNIT=6, FILE='INPTNME.TXT',STATUS='OLD')

      READ (6,16) FNAME
   16 FORMAT (A12)

      OPEN (UNIT=3, FILE='TITLE.TXT',STATUS='OLD')
      OPEN (UNIT=5, FILE='A.TXT',STATUS='OLD')
      OPEN (UNIT=10, FILE='B.TXT',STATUS='OLD')
      OPEN (UNIT=15, FILE='C.TXT',STATUS='OLD')
      OPEN (UNIT=13, FILE='DYN.TXT',STATUS='OLD')
      OPEN (UNIT=20, FILE='D.TXT',STATUS='OLD')
      OPEN (UNIT=25, FILE='E.TXT',STATUS='OLD')
      OPEN (UNIT=30, FILE='F.TXT',STATUS='OLD')
      OPEN (UNIT=35, FILE='G.TXT',STATUS='OLD')
      OPEN (UNIT=40, FILE='H.TXT',STATUS='OLD')
      OPEN (UNIT=45, FILE='I.TXT',STATUS='OLD')
      OPEN (UNIT=50, FILE='J.TXT',STATUS='OLD')
      OPEN (UNIT=55, FILE=FNAME,STATUS='UNKNOWN')

* Call Subroutines

      CALL INPUTA(A,TPRINT,NOSEG,MNSEG)
      CALL INPUTB(B,TPRINT,NOSEG,MODEL)
      CALL INPUTC(C,NOSEG)
      CALL INPUTD(D,TPRINT,MODEL)
      CALL INPUTE(E,TPRINT,MODEL)
      CALL INPUTF(F,MODEL,TPRINT,MNSEG)
      CALL INPUTG(G,NOSEG)
      CALL INPUTH(H,MODEL)
      CALL INPUTI(I)
      CALL INPUTJ(J,NOSEG)

      STOP
      END

******************** Subroutines


***Subroutine INPUTA:  Creates the A Block to the WASP5 input file
***The following defaults are set in this subroutine:
******Backward differencing is always used (ADFAC=0.0)
******A transport file is always generated (TFLG = 0)
******The first six segments are those which solns are displayed
*     on the screen (Record 5)
******The same maximum time step is used throughout the model-run
******The same print interval is used throughout the model-run
***This subroutine also reads the time the model will run (TPRINT)

      SUBROUTINE INPUTA(A,TPRINT,NOSEG,MNSEG)

      INTEGER NOSEG, NOSYS, ICFL, MFLAG, JMASS, NEGSLN, INTYP,
     & MODEL, TFLG, NH, NO, PO, CHL, CBOD, DO, ON, OP, ZMIN, ZHR,
     & MNSEG
      REAL A(14), ADFAC, ZDAY, DTS, TPRINT, PRTINV
      CHARACTER*80 TITLE
      CHARACTER*75 LEVEL

      READ (3,1) TITLE
      READ (5,*) A

* Declaring Variables

      MODEL = INT(A(1))
      NOSEG = INT(A(2))
      NOSYS = INT(A(3))
      ICFL = INT(A(4))
      MFLAG = INT(A(5))
      JMASS = INT(A(6))+1
      NEGSLN = INT(A(7))
      ZDAY = A(8)
      ZHR = INT(A(9))
      ZMIN = INT(A(10))
      INTYP = 1
      TFLG = 0
      ADFAC = 0.0
      DTS = A(11)
      PRTINV = A(13)
      TPRINT = A(12)
      MNSEG = INT(A(14))

      IF (MODEL .EQ. 0) THEN
          LEVEL = 'Simple Streeter-Phelps with SOD'
          NH = 1
          NO = 1
          PO = 1
          CHL = 1
          CBOD = 0
          DO = 0
          ON = 1
          OP = 1
      END IF

* Format statements

    1    FORMAT (A80)
    2    FORMAT (A5,1X,A)
    3    FORMAT (11A5,A25)
    4    FORMAT (7I5,2F5.0,I3,I2,I5)
    5    FORMAT (8I5)
    6    FORMAT (I5)
    7    FORMAT (2(F10.1, F10.1))
    9    FORMAT (F10.5,F10.1)

* Writing the input file

      WRITE (55,1) TITLE
      WRITE (55,2) "EUTRO ",LEVEL
      WRITE (55,3) "NSEG","NSYS","ICRD","MFLG","IDMP","NSLN",
     & "INTY","ADFC","DD","HHMM","TFLG","A: MODEL OPTIONS"
      WRITE (55,4) NOSEG, NOSYS, ICFL, MFLAG, JMASS, NEGSLN,
     & INTYP, ADFAC, ZDAY, ZHR, ZMIN, TFLG
      WRITE (55,5) 1,2,3,4,5,6
      WRITE (55,6) 1
      WRITE (55,9) DTS,TPRINT
      WRITE (55,6) 2
      WRITE (55,7) PRTINV,0.,PRTINV,TPRINT
      WRITE (55,5) NH,NO,PO,CHL,CBOD,DO,ON,OP

      RETURN
      END

***Subroutine INPUTB:  Creates the B Block to the WASP5 input file
***The following defaults are set in this subroutine:
******All exchanges are steady state
******The scaling and conversion factors are set to 1.0 (Areas should
*     be in m^2 and exchange coeff. in m^2/sec)
```

```
      SUBROUTINE INPUTB(B,TPRINT,NOSEG,MODEL)

      INTEGER MODEL,NRFLD,NTEX,NH,NO,PO,CHL,CBOD,DO,ON,OP,NORS,
     + TEMP,S, N, COUNT
      REAL B(100,5),TPRINT


      COUNT = 1
      DO WHILE (TEMP2 .NE. 555)
         READ (10,*) B(COUNT,1)
         TEMP2 = B(COUNT,1)
         COUNT = COUNT + 1
      END DO

      REWIND (UNIT = 10)
      N = COUNT-2
      DO I=1,N
              READ (10,*) (B(I,J),J=1,5)
      END DO

      MODEL = INT(B(1,1))
      NRFLD = INT(B(1,2))

      IF (MODEL.EQ.0) THEN
          NH = 1
          NO = 1
          PO = 1
          CHL = 1
          CBOD = 0
          DO = 0
          ON = 1
          OP = 1
      END IF

      WRITE (55,10) NRFLD,"B:EXCHANGES"

   10 FORMAT (I5,5X,A)
   11 FORMAT (I5,2F10.1)
   12 FORMAT (I5)
   13 FORMAT (2F10.1,2I5)
   14 FORMAT (4(F10.3,F10.1))
   15 FORMAT (8I5)

      NTEX = 1
      TEMP = 2
      DO I = TEMP, N-1
          IF (B(I,1).EQ.B(I+1,1)) THEN
          GO TO 500
          ELSE
          NTEX = NTEX + 1
          END IF
  500 END DO
```

```
      WRITE (55,11) NTEX,1.0,1.0

      DO S = 1,NTEX

        NORS = 1
          DO I = TEMP,N-1
              IF (B(I,1) .EQ. B(I+1,1)) THEN
              NORS = NORS + 1
              ELSE
              GO TO 510
              END IF
          END DO

  510   WRITE (55,12) NORS

          DO I = TEMP,TEMP+NORS-1
              WRITE (55,13) B(I,2),B(I,3),INT(B(I,4)),INT(B(I,5))
          END DO
        WRITE (55,12) 2
        WRITE (55,14) B(TEMP,1),0.0,B(TEMP,1),TPRINT
        TEMP = TEMP + NORS

      END DO

      WRITE (55,15) NH,NO,PO,CHL,CBOD,DO,ON,OP
      RETURN
      END

***Subroutine INPUTC:  Creates the C Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1.0 (Volumes should
*     be in m^3)
******The hydraulic coefficients used to calculate reaeration and
*     volitilization do not spatially vary

      SUBROUTINE INPUTC(C,NOSEG)

      INTEGER NOSEG, N
      REAL C((NOSEG+5),4)

      N = NOSEG + 5

      DO I=1,N
              READ (15,*) (C(I,J),J=1,4)
      END DO

   20 FORMAT (2I5,F10.4,5X,A)
   21 FORMAT (2F10.1)
   22 FORMAT (3I10,F10.1,F10.4,3F10.1)

      WRITE (55,20) INT(C(1,1)),INT(C(1,2)),C(1,3),"C:VOLUMES"
      WRITE (55,21) 1.0,1.0

      DO I = 6,N
          WRITE (55,22) INT(C(I,1)),INT(C(I,2)),INT(C(I,3)),C(I,4),
     &                  C(2,1),C(3,1),C(4,1),C(5,1)
      END DO

      RETURN
      END

***Subroutine INPUTD:  Creates the D Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1.0 (Flows should
*     be in m^3/sec)
******The number of flow fields is set to 1:  Water column only (no pore
*     water flows
******The flow is steady state


      SUBROUTINE INPUTD(D,TPRINT,MODEL)

      INTEGER  I,IQOPT,NFIELD,K,J,MODEL,H,FLW,
     & NH,NO,PO,CHL,CBOD,DO,ON,OP,W,COUNT,X
      REAL D(300,2),TPRINT,TEMP
      CHARACTER*12 HYDFILE

      IF (MODEL.EQ.0) THEN
          NH = 1
          NO = 1
          PO = 1
          CHL = 1
          CBOD = 0
          DO = 0
          ON = 1
          OP = 1
      END IF

      NFIELD = 1
      FLW = 0
      N = 2
      TEMP = 0.0
      DO WHILE (TEMP.NE.555.0)
          READ (20,*) (D(N,I),I=1,2)
          IF (D(N,2).EQ.999.0) THEN
              FLW = FLW+1
              PRINT *, FLW
          END IF
          TEMP = D(N,1)
          N = N+1
      END DO
      REWIND (UNIT = 20)

      DO I = 1,1
           READ(20,*) (D(I,J),J=1,2)
      END DO

      READ (13,35) HYDFILE
      IQOPT = INT(D(1,1))


      IF (IQOPT .EQ. 3) THEN
          WRITE (55,30) IQOPT,NFIELD,HYDFILE,"D:FLOWS"
          RETURN
      ELSE
          WRITE (55,30) IQOPT,NFIELD,HYDFILE,"D:FLOWS"
          WRITE (55,31) FLW, 1.0, 1.0
      END IF

      Z = 1
      X = 2
      TEMP = 0.0
      W = 1
      DO WHILE (TEMP.NE.555.0)
      COUNT = 0
      DO I = X,300
          READ(20,*) (D(I,J),J=1,2)
          IF (D(I,2).EQ.999) THEN
              GO TO 800
          ELSE IF (D(I,1).EQ.555.0) THEN
              GO TO 810
          ELSE
              COUNT = COUNT + 1
              W = W+1
          END IF
      END DO


  800 WRITE (55,32) COUNT

      DUMMY1 = INT(COUNT/4)
      DUMMY2 = COUNT - DUMMY1*4

      IF (DUMMY1.EQ.0) THEN
          DUMMY3 = X
      END IF

      DO K = X,(4*DUMMY1)+X-1,4
          WRITE (55,33) 1.0,INT(D(K,1)),INT(D(K,2)),
     &     1.0,INT(D(K+1,1)),INT(D(K+1,2)),
     &     1.0,INT(D(K+2,1)),INT(D(K+2,2)),
     &     1.0,INT(D(K+3,1)),INT(D(K+3,2))
           DUMMY3 = K+4
      END DO

      IF (DUMMY2.NE.0) THEN
      WRITE (55,33)(1.0,INT(D(H,1)),INT(D(H,2)),H=DUMMY3,DUMMY2+
     & DUMMY3-1)
      END IF

      WRITE (55,32) 2
```

```
      WRITE (55,34) D(W+Z,1),0.0,D(W+Z,1),TPRINT

      X = COUNT+X+1
      Z = Z+1
  810 TEMP = D(I,1)
      END DO

   30 FORMAT (2I5,A12,5X,A)
   31 FORMAT (I5,2F10.1)
   32 FORMAT (I5)
   33 FORMAT (4(F10.2,2I5))
   34 FORMAT (4F10.2)
   35 FORMAT (A12)
   36 FORMAT (8I5)

      WRITE (55,36) NH,NO,PO,CHL,CBOD,DO,ON,OP

      RETURN
      END

***Subroutine INPUTE:  Creates the E Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1.0 (Boundary conditions
*     should be in mg/L)
******Only BOD and DO are considered
******The bc's are steady state


      SUBROUTINE INPUTE(E,TPRINT,MODEL)

      INTEGER  NOBC,I,MODEL,NH,NO,PO,CHL,CBOD,DO,ON,OP,
     & N
      REAL E(100,2),TPRINT


      DO I = 1,1
           READ(25,*) (E(I,J),J=1,2)
      END DO

      NOBC = INT(E(1,1))
      N = 2*NOBC + 1

      IF (NOBC .EQ. 0) THEN
          WRITE (55,40) 0,"*","E:BOUNDARIES"
          RETURN
      END IF

      DO I = 2,N
          READ(25,*) (E(I,J),J=1,2)
      END DO

   40 FORMAT (I10,2X,A5,5X,A)
   41 FORMAT (I10,2X,A)
   42 FORMAT (2I5)
   43 FORMAT (2(2F10.2))

      IF (MODEL.EQ.0) THEN
          NH = 0
          NO = 0
          PO = 0
          CHL = 0
          CBOD = NOBC
          DO = NOBC
          ON = 0
          OP = 0
      END IF

      WRITE (55,40) NH,"NH3","E:BOUNDARIES"
      WRITE (55,41) NO,"NO3"
      WRITE (55,41) PO,"PO4"
      WRITE (55,41) CHL,"CHLa"
      WRITE (55,41) CBOD,"CBOD"
      WRITE (55,43) 1.0,1.0
      DO I = 2,NOBC+1
          WRITE (55,42) INT(E(I,1)),2
          WRITE (55,43) E(I,2),0.0,E(I,2),TPRINT
      END DO
      WRITE (55,41) DO,"DO"
      WRITE (55,43) 1.0,1.0
      DO I = NOBC+2,2*NOBC+1
          WRITE (55,42) INT(E(I,1)),2
          WRITE (55,43) E(I,2),0.0,E(I,2),TPRINT
      END DO
      WRITE (55,41) ON,"ON"
      WRITE (55,41) OP,"OP"

      RETURN
      END

***Subroutine INPUTF:  Creates the F Block to the WASP5 input file
***The following defaults are set in this subroutine:
******All point sources are steady state
******At the present time, all non-point sources are also steady state
******Loads are in kg/day


      SUBROUTINE INPUTF(F,MODEL,TPRINT,MNSEG)

      INTEGER  MODEL,NH,NO,PO,CHL,CBOD,DO,ON,OP,N,I,J,X,W,
     & MNSEG
      REAL F(25,2),TPRINT,TEMP(26,2),LOAD,NPS

      G = 2*MNSEG+1
      DO I = 1,G
          READ (30,*) (F(I,J),J=1,2)
      END DO

      N = INT(F(1,1))
```

```
      IF (MODEL.EQ.0) THEN
          NH = 0
          NO = 0
          PO = 0
          CHL = 0
          CBOD = N
          DO = 0
          ON = 0
          OP = 0
      END IF


   60 FORMAT (I10,5X,A)
   61 FORMAT (2F10.1)
   62 FORMAT (2I5)
   63 FORMAT (2(2F10.2))

      WRITE (55,60) NH,"NH3      F:LOADS"
      WRITE (55,60) NO, "NO3"
      WRITE (55,60) PO, "PO4"
      WRITE (55,60) CHL,"CHLa"
      WRITE (55,60) CBOD,"CBOD"
      WRITE (55,61) 1.0,1.0

      X = 1
      DO J = 1,N
      DO I = 2,N+1
          W = INT(F(I,1))
          IF (W.EQ.X) THEN
              TEMP(I,1) = F(I,1)
              TEMP(I,2) = F(I,2)
              GO TO 700
          END IF
      END DO
  700 X = X+1
      END DO

      DO I = 2,N+1
          NPS = F(I+N,2)
          LOAD = TEMP(I,2) + NPS
          WRITE (55,62) INT(TEMP(I,1)),2
          WRITE (55,63) LOAD,0.0,LOAD,TPRINT
      END DO

      WRITE (55,60) DO,"DO"
      WRITE (55,60) ON,"ON"
      WRITE (55,60) OP,"OP"
      WRITE (55,60) 0


      RETURN
      END
```

```
***Subroutine INPUTG:  Creates the G Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The scaling and conversion factors are set to 1.0
*       Temperatures are in °C
*       Sediment Oxygen Demand are in g/m^2-day
*       Salinity is in ppt
******The theta used to correct SOD for temperature does not
*     spatially vary


      SUBROUTINE INPUTG(G,NOSEG)

      INTEGER NOSEG
      REAL G((NOSEG+1),4), SODTA

      DO I = 1,NOSEG+1
           READ(35,*) (G(I,J),J=1,4)
      END DO

      SODTA = G(1,1)

   50 FORMAT (I10,5X,A)
   51 FORMAT (4(A5,I5,F10.3))
   52 FORMAT (I10)

      WRITE (55,50) 4,"G:PARAMETERS"
      WRITE (55,51) "TMPSG",3,1.0,"SOD1D",9,1.0,"SODTA",
     & 12,1.0,"SAL",2,1.0

      DO I = 2,NOSEG+1
          WRITE (55,52) INT(G(I,1))
          WRITE (55,51) "TMPSG",3,G(I,2),"SOD1D",9,G(I,3),"SODTA",
     & 12,SODTA,"SAL",2,G(I,4)
      END DO

      RETURN
      END

***Subroutine INPUTH:  Creates the H Block to the WASP5 input file
***The following defaults are set in this subroutine:
******The reareartion rate (/day) does not spatially vary
******The deoxygenation coefficient (/day) does not spatially vary


      SUBROUTINE INPUTH(H,MODEL)

      INTEGER  MODEL,NH,NO,PO,CHL,CBOD,DO,ON,OP
      REAL H(2),KD,KA


      READ(40,*) H


      KD = H(1)
      KA = H(2)

   70 FORMAT (10X,A)
   71 FORMAT (A10,I10)
   72 FORMAT (A10,I10,F10.2)

      IF (MODEL.EQ.0) THEN
          NH = 0
          NO = 0
          PO = 0
          CHL = 0
          CBOD = 1
          DO = 1
          ON = 0
          OP = 0
      END IF

      WRITE (55,70) "H:CONSTANTS"
      WRITE (55,71) "GLOBALS",0
      WRITE (55,71) "NH3",NH
      WRITE (55,71) "NO3",NO
      WRITE (55,71) "PO4",PO
      WRITE (55,71) "CHLa",CHL
      WRITE (55,71) "CBOD",CBOD
      WRITE (55,72) "deoxygentation",1
      WRITE (55,72) "KD",71,KD
      WRITE (55,71) "DO",DO
      WRITE (55,72) "oxygenation",1
      WRITE (55,72) "K2",82,KA
      WRITE (55,71) "ON",ON
      WRITE (55,71) "OP",OP

      RETURN
      END

***Subroutine INPUTI:  Creates the I Block to the WASP5 input file
***The following defaults are set in this subroutine:
******Since this model is considered steady state (right now), there
*     are no time functions presently input into the model


      SUBROUTINE INPUTI(I)

      REAL I(1)

      READ (45,*) I

      WRITE (55,80) INT(I(1)),"I:TIME FUNCTIONS"
   80 FORMAT (I10,5X,A)

      RETURN
      END

***Subroutine INPUTJ:  Creates the J Block to the WASP5 input file
```

```
***The following defaults are set in this subroutine:
******This program, at present only considers the bod and do i.c.'s
******The dissolved fraction of BOD is set at 0.5
******The dissolved fraction of DO is always 1.0
******The maximum value of all variables is set at 1.0e8
******Solids Field 3 is what transports the system in its particulate
*     form  for BOD and Solids Field 5 is used for DO
******The densities of BOD and DO are set at 1.0 -- EUTRO does not
*     use those numbers
******All initial conditions are in mg/L


      SUBROUTINE INPUTJ(I,NOSEG)

      INTEGER  NOSEG,J
      REAL I(NOSEG*8+8,2),DENBOD,DENDO,MAX

      DO J = 1,NOSEG*8+8
          READ (50,*) (I(J,K),K=1,2)
      END DO

      MAX = 100000000.0
      DENBOD = 1.0
      DENDO = 1.0

   90 FORMAT (A10,30X,I5,F5.1,E10.2,A)
   91 FORMAT (3(I5,2F10.2))

      DUMMY1 = INT(NOSEG/3)
      DUMMY2= NOSEG - 3*DUMMY1

      WRITE (55,90) "NH3",3,1.2,MAX,"J:INITIAL CONDITIONS"
      DO J = 2,3*DUMMY1+1,3
          WRITE (55,91) INT(I(J,1)),I(J,2),I(1,1),
     &         INT(I(J+1,1)),I(J+1,2),I(1,1),
     &         INT(I(J+2,1)),I(J+1,2),I(1,1)
      DUMMY3 = J+3
      END DO
      WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
     &    +DUMMY3-1)

      X = DUMMY3+DUMMY2+1
      WRITE (55,90) "NO3",5,1.2,MAX
      DO J = X,6*DUMMY1+DUMMY2+2,3
          WRITE (55,91) INT(I(J,1)),I(J,2),I(NOSEG+2,1),
     &         INT(I(J+1,1)),I(J+1,2),I(NOSEG+2,1),
     &         INT(I(J+2,1)),I(J+1,2),I(NOSEG+2,1)
      DUMMY3 = J+3
      END DO
      WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
     &    +DUMMY3-1)


      X = DUMMY3+DUMMY2+1
```

```
 WRITE (55,90) "PO4",5,1.2,MAX
 DO J = X,9*DUMMY1+2*DUMMY2+3,3
     WRITE (55,91) INT(I(J,1)),I(J,2),I(2*NOSEG+3,1),
&         INT(I(J+1,1)),I(J+1,2),I(2*NOSEG+3,1),
&         INT(I(J+2,1)),I(J+1,2),I(2*NOSEG+3,1)
 DUMMY3 = J+3
 END DO
 WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
&    +DUMMY3-1)

 X = DUMMY3+DUMMY2+1
 WRITE (55,90) "PHYT",4,1.2,MAX
 DO J = X,12*DUMMY1+3*DUMMY2+4,3
     WRITE (55,91) INT(I(J,1)),I(J,2),I(3*NOSEG+4,1),
&         INT(I(J+1,1)),I(J+1,2),I(3*NOSEG+4,1),
&         INT(I(J+2,1)),I(J+1,2),I(3*NOSEG+4,1)
 DUMMY3 = J+3
 END DO
 WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
&    +DUMMY3-1)

 X = DUMMY3+DUMMY2+1
 WRITE (55,90) "CBOD",3,DENBOD,MAX
 DO J = X,15*DUMMY1+4*DUMMY2+5,3
     WRITE (55,91) INT(I(J,1)),I(J,2),I(4*NOSEG+5,1),
&         INT(I(J+1,1)),I(J+1,2),I(4*NOSEG+5,1),
&         INT(I(J+2,1)),I(J+1,2),I(4*NOSEG+5,1)
 DUMMY3 = J+3
 END DO
 WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
&    +DUMMY3-1)

 X = DUMMY3+DUMMY2+1
 WRITE (55,90) "DO",5,DENDO,MAX
 DO J = X,18*DUMMY1+5*DUMMY2+6,3
     WRITE (55,91) INT(I(J,1)),I(J,2),I(5*NOSEG+6,1),
&         INT(I(J+1,1)),I(J+1,2),I(5*NOSEG+6,1),
&         INT(I(J+2,1)),I(J+1,2),I(5*NOSEG+6,1)
 DUMMY3 = J+3
 END DO
 WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
&    +DUMMY3-1)

 X = DUMMY3+DUMMY2+1
 WRITE (55,90) "ON",3,1.2,MAX
 DO J = X,21*DUMMY1+6*DUMMY2+7,3
     WRITE (55,91) INT(I(J,1)),I(J,2),I(6*NOSEG+7,1),
&         INT(I(J+1,1)),I(J+1,2),I(6*NOSEG+7,1),
&         INT(I(J+2,1)),I(J+1,2),I(6*NOSEG+7,1)
 DUMMY3=J+3
 END DO
 WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
&    +DUMMY3-1)
```

```
      X = DUMMY3+DUMMY2+1
      WRITE (55,90) "OP",3,1.2,MAX
      DO J = X,24*DUMMY1+7*DUMMY2+8,3
          WRITE (55,91) INT(I(J,1)),I(J,2),I(7*NOSEG+8,1),
     &         INT(I(J+1,1)),I(J+1,2),I(7*NOSEG+8,1),
     &         INT(I(J+2,1)),I(J+1,2),I(7*NOSEG+8,1)
      DUMMY3 = J+3
      END DO
      WRITE (55,91) (INT(I(J,1)),I(J,2),I(1,1),J=DUMMY3,DUMMY2
     &    +DUMMY3-1)


      RETURN
      END
```

Appendix C-5
FORTRAN Programs Used to Format WASP5 Output for Avenue Processing

Program: calout.for
Formats TOXI5 output

```
******************************************************
*  Program:   calout.for
*  Purpose:   This program will take WASP5 output
*             (specifically TOXI5, Level One Complexity
*             and create an array, written to a txt
*             File (salinity.txt).  This text file will
*             then be read by Avenue and imported into
*             ArcView for viewing with Charts and graphs
*  Hardware:  IBM Pentium 100
*  Software:  Microsoft FORTRAN
*  Author:    Jennifer Benaman
*             Graduate Research Assistant,
*             University of Texas at Austin
*  Date:      June, 1996
********************************************************

*  Program Declaration
      PROGRAM CALOUT

*  Variable Declaration
      REAL TPRINT,DTS,PRINTINV,INT(4),OUT(20000,6),
     & TEMP(20000,100)

      INTEGER NSEG,N,M
      CHARACTER*12 FNAME

*     TPRINT:         the final time (days) the model was run
*     DTS:            the model maximum time step (days)
*     PRINTINV:       the print interval (days) that the output
*                     was printed to the file
*     NSEG:           the total number of segments in the system
*     OUT(20000,6):   an array that holds the initial output file
*     TEMP(20000,100):an array that holds the values for the text
*                     file for the final output

      OPEN (UNIT=6, FILE='OUTNME.TXT',STATUS='OLD')

      READ (6,16) FNAME
   16 FORMAT (A12)

      OPEN (UNIT=30,FILE="START.TXT",STATUS="OLD")
      OPEN (UNIT=40,FILE=FNAME,STATUS="OLD")
      OPEN (UNIT=50,FILE="SALINITY.TXT",STATUS="UNKNOWN")

      CALL INITIAL(TPRINT,DTS,PRINTINV,NSEG,INT,N,M)
      CALL OUTPUT(NSEG,OUT,TEMP,N,M)

      STOP
      END

***************************Subroutines

*****Subroutine: initial
******This subroutine will read the intial information from
******a text file created by Avenue

      SUBROUTINE INITIAL(TPRINT,DTS,PRINTINV,NSEG,I,N,M)

      REAL TPRINT,DTS,PRINTINV,I(4)
      INTEGER NSEG

      READ (30,*) I

      TPRINT = I(1)
      DTS = I(2)
      PRINTINV = I(3)
      NSEG = INT(I(4))

      N = INT((TPRINT/PRINTINV)+1)
      M = INT(4*NSEG*(TPRINT/PRINTINV+1)+22)

      RETURN
      END

*****Subroutine: output
******This subroutine will take the output file generated by TOXI5 and
******just read the salinity values at each time step and create a
******new array which has just the time steps and salinity for each
******segment

      SUBROUTINE OUTPUT(NSEG,OUT,TEMP,N,M)

      INTEGER NSEG,N,M,T,X
      REAL TEMP(N,NSEG+1),OUT(M,6)
      CHARACTER*20 JUNK

      DO I = 1,22
       READ (40,10) JUNK
      END DO
   10 FORMAT (A20)
   20 FORMAT (F6.0,F10.0)
   30 FORMAT (6E11.3)
   40 FORMAT (F10.3,50(F7.3))

      X = 1
      DO I = 23,M,4
          READ (40,20) (OUT(I,J),J=1,2)
          READ (40,30) ((OUT(K,J),J=1,6),K=I+1,I+3)
          X = X+1
      END DO

      T = 1
      DO I = 23,M,4*NSEG
          TEMP(T,1) = OUT(I,2)
          DO J = 2,NSEG+1
            IF (J.EQ.2) THEN
            TEMP(T,J) = OUT(I+2,1)/1000000
            ELSE
            TEMP(T,J) = OUT((I+2+4*(J-2)),1)/1000000
            END IF
          END DO
      WRITE (50,40) (TEMP(T,J),J=1,NSEG+1)
      T = T+1
      END DO

      RETURN
      END
```

Program: modout.for
Formats EUTRO5 output

```
*********************************k*********************
*  Program:   modout.for
*  Purpose:   This program will take WASP5 output
*             (specifically EUTRO5, Level One Complexity
*             and create an array, written to txt
*             Files (bod.txt and do.txt).  These text file will
*             then be read by Avenue and imported into
*             ArcView for viewing with Charts and graphs
*  Hardware:  IBM Pentium 100
*  Software:  Microsoft FORTRAN
*  Author:    Jennifer Benaman
*             Graduate Research Assistant,
*             University of Texas at Austin
*  Date:      June, 1996
*******************************************************

*  Program Declaration
      PROGRAM MODOUT

*  Variable Declaration
      REAL TPRINT,DTS,PRINTINV,INT(4),OUT(20000,6),
     & TEMP1(20000,100),TEMP2(20000,100)

      INTEGER NSEG,N,M
      CHARACTER*12 FNAME

*     TPRINT:         the final time (days) the model was run
*     DTS:            the model maximum time step (days)
*     PRINTINV:       the print interval (days) that the output
*                     was printed to the file
*     NSEG:           the total number of segments in the system
*     OUT(20000,6):   an array that holds the initial output file
*     TEMP1(20000,100):an array that holds the do values for the text
*                     file for the final output (do.txt)
*     TEMP2(20000,100):an array that holds the bod values for the text
*                     file for the final output(bod.txt)

      OPEN (UNIT=6, FILE="EOUTNME.TXT",STATUS="OLD")
      READ (6,16) FNAME
   16 FORMAT (A12)

      OPEN (UNIT=30,FILE="ESTART.TXT",STATUS="OLD")
      OPEN (UNIT=40,FILE=FNAME,STATUS="OLD")
      OPEN (UNIT=50,FILE="DO.TXT",STATUS="UNKNOWN")
      OPEN (UNIT=60,FILE="BOD.TXT",STATUS="UNKNOWN")

      CALL INITIAL(TPRINT,DTS,PRINTINV,NSEG,INT,N,M)
      CALL OUTPUT(NSEG,OUT,TEMP1,TEMP2,N,M)
      STOP
      END

***************************Subroutines

*****Subroutine: initial
******This subroutine will read the intial information from
******a text file created by Avenue

      SUBROUTINE INITIAL(TPRINT,DTS,PRINTINV,NSEG,I,N,M)

      REAL TPRINT,DTS,PRINTINV,I(4)
      INTEGER NSEG

      READ (30,*) I

      TPRINT = I(1)
      DTS = I(2)
      PRINTINV = I(3)
      NSEG = INT(I(4))

      N = INT((TPRINT/PRINTINV)+1)
      M = INT(8*NSEG*(TPRINT/PRINTINV+1)+48)

      RETURN
      END

*****Subroutine: output
******This subroutine will take the output file generated by EUTRO5 and
******just read the bod and do values at each time step and create a
******two new arrays which have just the time steps and do and bod for each
******segment

      SUBROUTINE OUTPUT(NSEG,OUT,TEMP1,TEMP2,N,M)

      INTEGER NSEG,N,M,T,X
      REAL TEMP1(N,NSEG+1),TEMP2(N,NSEG+1),OUT(M,6)
      CHARACTER*20 JUNK

      DO I = 1,48
       READ (40,10) JUNK
      END DO

   10 FORMAT (A20)
   20 FORMAT (F6.0,F12.0)
   30 FORMAT (6E11.3)
   40 FORMAT (F10.3,50(F7.3))

      DO I = 49,M,8
          READ (40,20) (OUT(I,J),J=1,2)
          READ (40,30) ((OUT(K,J),J=1,6),K=I+1,I+7)
      END DO

      T = 1
      DO I = 49,M,8*NSEG
          TEMP1(T,1) = OUT(I,2)
          TEMP2(T,1) = OUT(I,2)
          DO J = 2,NSEG+1
            IF (J.EQ.2) THEN
            TEMP1(T,J) = OUT(I+1,5)
            TEMP2(T,J) = OUT(I+3,2)
            ELSE
            TEMP1(T,J) = OUT((I+1+8*(J-2)),5)
            TEMP2(T,J) = OUT((I+3+8*(J-2)),2)
            END IF
          END DO
      WRITE (50,40) (TEMP1(T,J),J=1,NSEG+1)
      WRITE (60,40) (TEMP2(T,J),J=1,NSEG+1)
      T = T+1
      END DO

      RETURN
      END
```

Appendix D
WASP5 Input Files

Appendix D-1
Input File for WASP5 for Model Calibration,
Created by Avenue and FORTRAN Formatting Program (calgen.for)

```
Salinity Calibration
TOXI5 Calibration with Salinity
 NSEG NSYS ICRD MFLG IDMP NSLN INTY ADFC   DD HHMM TFLG        A: MODEL OPTIONS
   17    1    0    0    1    0    1   0.   0.  0 0    0
    1    2    3    4    5    6
    1
    .05000     100.0
    2
        .5        .0        .5     100.0
    0    1    1    1    1    1    1
    1     B:EXCHANGES
    5       1.0       1.0
    7
    1625.8    3093.5    6    7
    2471.2    2947.1    7    8
    1625.8    3867.8    5    6
    1625.8    3388.5    1    2
    1625.8    3334.9    4    5
    1625.8    2764.2    2    3
    1625.8    2310.7    3    4
    2
     704.5        .0     704.5     100.0
    7
     465.4    5642.7   10    1
     168.2    1609.8   12    2
     505.4    1609.8   11    3
     185.8    1609.8   13    4
     528.6    1609.8   14    5
     717.2    1609.8   15    6
     260.1    1609.8   16    8
    2
     119.9        .0     119.9     100.0
    1
    2471.2    3311.4   17    8
    2
     704.5        .0     704.5     100.0
    1
    2471.2    1609.3   17    0
    2
     959.3        .0     959.3     100.0
    8
    3404.2       4.0    8    9
    3697.1       3.1    6    9
    2489.9       4.0    7    9
    4038.5       4.6    5    9
    3238.5       4.6    1    9
    2631.4       3.1    4    9
```

```
 3538.5       4.6    2    9
 1989.9       4.6    3    9
 2
     .0        .0        .0     100.0
 0    1    1    1    1    1    1    1
 2    0    1.0000     C:VOLUMES
    1.0       1.0
      8         9         1 8412370.0     .0040        .4       1.2        .6
      6         9         1 6010670.0     .0040        .4       1.2        .6
      7         9         1 6153160.0     .0040        .4       1.2        .6
      5         9         1 6565760.0     .0040        .4       1.2        .6
      1         9         1 5265120.0     .0040        .4       1.2        .6
      4         9         1 4278080.0     .0040        .4       1.2        .6
      2         9         1 5752860.0     .0040        .4       1.2        .6
      3         9         1 3235260.0     .0040        .4       1.2        .6
     10         0         1 3745030.0     .0040        .4       1.2        .6
     12         0         1  541385.0     .0040        .4       1.2        .6
     11         0         1 1626730.0     .0040        .4       1.2        .6
     13         0         1  598034.0     .0040        .4       1.2        .6
     14         0         1 1701400.0     .0040        .4       1.2        .6
     15         0         1 2308450.0     .0040        .4       1.2        .6
     16         0         1  837184.0     .0040        .4       1.2        .6
     17         0         1 7954050.0     .0040        .4       1.2        .6
      9         0         3    2503.8     .0040        .4       1.2        .6
 2    1nothing          D:FLOWS
 8       1.0       1.0
11
   1.00    0   10      1.00   10    1      1.00    1    2      1.00    2    3
   1.00    3    4      1.00    4    5      1.00    5    6      1.00    6    7
   1.00    7    8      1.00    8   17      1.00   17    0
 2
  14.35       .00     14.35    100.00
10
   1.00    0   12      1.00   12    2      1.00    2    3      1.00    3    4
   1.00    4    5      1.00    5    6      1.00    6    7      1.00    7    8
   1.00    8   17      1.00   17    0
 2
   5.70       .00      5.70    100.00
 9
   1.00    0   11      1.00   11    3      1.00    3    4      1.00    4    5
   1.00    5    6      1.00    6    7      1.00    7    8      1.00    8   17
   1.00   17    0
 2
   3.54       .00      3.54    100.00
 8
   1.00    0   13      1.00   13    4      1.00    4    5      1.00    5    6
   1.00    6    7      1.00    7    8      1.00    8   17      1.00   17    0
```

```
    2
      1.00       .00      1.00    100.00
    7
      1.00    0   14      1.00   14    5      1.00    5    6      1.00    6    7
      1.00    7    8      1.00    8   17      1.00   17    0
    2
      2.06       .00      2.06    100.00
    6
      1.00    0   15      1.00   15    6      1.00    6    7      1.00    7    8
      1.00    8   17      1.00   17    0
    2
      7.59       .00      7.59    100.00
    4
      1.00    0   16      1.00   16    8      1.00    8   17      1.00   17    0
    2
      1.20       .00      1.20    100.00
    2
      1.00    0   17      1.00   17    0
    2
     42.40       .00     42.40    100.00
    0    1    1    1    1    1    1    1
         8    SAL      E:BOUNDARIES
   1000.00      1.00
   10    2
      1.82       .00      1.82    100.00
   17    2
     10.90       .00     10.90    100.00
   12    2
```

```
        .20       .00       .20    100.00
   11    2
        .20       .00       .20    100.00
   13    2
        .20       .00       .20    100.00
   14    2
        .20       .00       .20    100.00
   15    2
        .20       .00       .20    100.00
   16    2
        .20       .00       .20    100.00
          0     F:LOADS -- NO LOADS
          0     NO NPS LOADS
          0     G:PARAMETERS -- NO PARAMETERS
           H:CONSTANTS
   GLOBALS         0
  SALINITY         1
   GENERAL         3
      PIXC       111       .00       KBW       141       .00
     MOLWT        81     78.50
          0                    I:TIME FUNCTIONS
       SAL                                  3   .0   .35E+05J:INITIAL CONDITIONS
    1       .00      1.00    2       .00      1.00    3       .00      1.00
    4       .00      1.00    5       .00      1.00    6       .00      1.00
    7       .00      1.00    8       .00      1.00    9       .00      1.00
   10   1820.00      1.00   11       .00      1.00   12       .00      1.00
   13       .00      1.00   14       .00      1.00   15       .00      1.00
   16       .00      1.00   17  10900.00      1.00
```

Appendix D-2
Input File for WASP5 for BOD/DO Model, Created by Avenue and FORTRAN Formatting Program (outgen.for)

Average Year Scenario

```
Average Year Conditions -- Initial
EUTRO Simple Streeter-Phelps with SOD
 NSEG NSYS ICRD MFLG IDMP NSLN INTY ADFC   DD HHMM TFLG         A: MODEL OPTIONS
   17    8    0    0    5    1    1   0.   0.  0 0    0
    1    2    3    4    5    6
    1
    .05000     100.0
    2
        .5        .0        .5     100.0
    1    1    1    1    0    0    1    1
    1     B:EXCHANGES
    5       1.0       1.0
    7
    1625.8    3093.5    6    7
    2471.2    2947.1    7    8
    1625.8    3867.8    5    6
```

```
  1625.8    3388.5    1    2
  1625.8    3334.9    4    5
  1625.8    2764.2    2    3
  1625.8    2310.7    3    4
 2
704.500        .0   704.500     100.0
 7
   465.4    5642.7   10    1
   168.2    1609.8   12    2
   505.4    1609.8   11    3
   185.8    1609.8   13    4
   528.6    1609.8   14    5
   717.2    1609.8   15    6
   260.1    1609.8   16    8
 2
119.900        .0   119.900     100.0
```

```
 1
 2471.2    3311.4   17    8
 2
704.500        .0   704.500     100.0
 1
 2471.2    1609.3   17    0
 2
959.300        .0   959.300     100.0
 8
 3404.2       4.0    8    9
 3697.1       3.1    6    9
 2489.9       4.0    7    9
 4038.5       4.6    5    9
 3238.5       4.6    1    9
 2631.4       3.1    4    9
 3538.5       4.6    2    9
 1989.9       4.6    3    9
 2
   .001        .0      .001     100.0
 1    1    1    1    0    0    1    1
 2    0    1.0000     C:VOLUMES
    1.0       1.0
      8         9         1 8412370.0     .0040        .4       1.2        .6
      6         9         1 6010670.0     .0040        .4       1.2        .6
      7         9         1 6153160.0     .0040        .4       1.2        .6
      5         9         1 6565760.0     .0040        .4       1.2        .6
      1         9         1 5265120.0     .0040        .4       1.2        .6
      4         9         1 4278080.0     .0040        .4       1.2        .6
      2         9         1 5752860.0     .0040        .4       1.2        .6
      3         9         1 3235260.0     .0040        .4       1.2        .6
     10         0         1 3745030.0     .0040        .4       1.2        .6
     12         0         1  541385.0     .0040        .4       1.2        .6
     11         0         1 1626730.0     .0040        .4       1.2        .6
     13         0         1  598034.0     .0040        .4       1.2        .6
     14         0         1 1701400.0     .0040        .4       1.2        .6
     15         0         1 2308450.0     .0040        .4       1.2        .6
     16         0         1  837184.0     .0040        .4       1.2        .6
     17         0         1 7954050.0     .0040        .4       1.2        .6
      9         0         3    2503.8     .0040        .4       1.2        .6
 2    1nothing          D:FLOWS
 8       1.0       1.0
11
   1.00    0   10      1.00   10    1      1.00    1    2      1.00    2    3
   1.00    3    4      1.00    4    5      1.00    5    6      1.00    6    7
   1.00    7    8      1.00    8   17      1.00   17    0
 2
  14.35       .00     14.35    100.00
10
   1.00    0   12      1.00   12    2      1.00    2    3      1.00    3    4
   1.00    4    5      1.00    5    6      1.00    6    7      1.00    7    8
   1.00    8   17      1.00   17    0
 2
   5.70       .00      5.70    100.00
 9
```

```
  1.00    0   11      1.00   11    3      1.00    3    4      1.00    4    5
  1.00    5    6      1.00    6    7      1.00    7    8      1.00    8   17
  1.00   17    0
2
  3.54       .00      3.54    100.00
8
  1.00    0   13      1.00   13    4      1.00    4    5      1.00    5    6
  1.00    6    7      1.00    7    8      1.00    8   17      1.00   17    0
2
  1.00       .00      1.00    100.00
7
  1.00    0   14      1.00   14    5      1.00    5    6      1.00    6    7
  1.00    7    8      1.00    8   17      1.00   17    0
2
  2.06       .00      2.06    100.00
6
  1.00    0   15      1.00   15    6      1.00    6    7      1.00    7    8
  1.00    8   17      1.00   17    0
2
  7.59       .00      7.59    100.00
4
  1.00    0   16      1.00   16    8      1.00    8   17      1.00   17    0
2
  1.20       .00      1.20    100.00
2
  1.00    0   17      1.00   17    0
2
 42.40       .00     42.40    100.00
 1    1    1    1    0    0    1    1
     0    NH3     E:BOUNDARIES
     0  NO3
     0  PO4
     0  CHLa
     8  CBOD
  1.00      1.00
10    2
  1.92       .00      1.92    100.00
12    2
  1.37       .00      1.37    100.00
11    2
  1.87       .00      1.87    100.00
13    2
  1.68       .00      1.68    100.00
14    2
  1.67       .00      1.67    100.00
15    2
  1.48       .00      1.48    100.00
16    2
  1.35       .00      1.35    100.00
17    2
  7.42       .00      7.42    100.00
     8  DO
  1.00      1.00
10    2
```

```
      3.03       .00      3.03    100.00
   12    2
      5.00       .00      5.00    100.00
   11    2
      5.00       .00      5.00    100.00
   13    2
      5.00       .00      5.00    100.00
   14    2
      5.00       .00      5.00    100.00
   15    2
      5.00       .00      5.00    100.00
   16    2
      5.00       .00      5.00    100.00
   17    2
      3.63       .00      3.63    100.00
         0  ON
         0  OP
         0     NH3      F:LOADS
         0     NO3
         0     PO4
         0     CHLa
         8     CBOD
       1.0       1.0
    1    2
   8010.42       .00   8010.42    100.00
    2    2
   3709.71       .00   3709.71    100.00
    3    2
   2080.65       .00   2080.65    100.00
    4    2
   2474.68       .00   2474.68    100.00
    5    2
   1432.67       .00   1432.67    100.00
    6    2
   4718.60       .00   4718.60    100.00
    7    2
    146.86       .00    146.86    100.00
    8    2
    973.81       .00    973.81    100.00
         0     DO
         0     ON
         0     OP
         0
         4     G:PARAMETERS
TMPSG    3     1.000SOD1D    9     1.000SODTA   12     1.000  SAL    2     1.000
         8
TMPSG    3    24.000SOD1D    9     1.500SODTA   12     1.065  SAL    2     9.850
         6
TMPSG    3    24.200SOD1D    9     1.500SODTA   12     1.065  SAL    2     9.450
         7
TMPSG    3    24.000SOD1D    9     1.500SODTA   12     1.065  SAL    2     9.850
         5
TMPSG    3    26.400SOD1D    9     1.500SODTA   12     1.065  SAL    2     9.960
         1
```

```
TMPSG    3    28.000SOD1D    9     1.500SODTA   12     1.065  SAL    2     5.820
         4
TMPSG    3    23.800SOD1D    9     1.500SODTA   12     1.065  SAL    2     7.870
         2
TMPSG    3    23.800SOD1D    9     1.500SODTA   12     1.065  SAL    2     7.870
         3
TMPSG    3    23.800SOD1D    9     1.500SODTA   12     1.065  SAL    2     7.870
        10
TMPSG    3    23.600SOD1D    9     1.500SODTA   12     1.065  SAL    2     1.820
        12
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .200
        11
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .200
        13
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .200
        14
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .200
        15
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .200
        16
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .200
        17
TMPSG    3    25.300SOD1D    9     1.500SODTA   12     1.065  SAL    2    10.900
         9
TMPSG    3    20.000SOD1D    9     1.500SODTA   12     1.065  SAL    2      .000
          H:CONSTANTS
   GLOBALS         0
       NH3         0
       NO3         0
       PO4         0
      CHLa         0
      CBOD         1
deoxygenta         1
        KD        71       .10
        DO         1
oxygenatio         1
        K2        82       .10
        ON         0
        OP         0
         0     I:TIME FUNCTIONS
       NH3                                 3  1.2   .10E+09J:INITIAL CONDITIONS
    1       .00      1.00    2       .00      1.00    3       .00      1.00
    4       .00      1.00    5       .00      1.00    6       .00      1.00
    7       .00      1.00    8       .00      1.00    9       .00      1.00
   10       .00      1.00   11       .00      1.00   12       .00      1.00
   13       .00      1.00   14       .00      1.00   15       .00      1.00
   16       .00      1.00   17       .00      1.00
       NO3                                 5  1.2   .10E+09
    1       .00      1.00    2       .00      1.00    3       .00      1.00
    4       .00      1.00    5       .00      1.00    6       .00      1.00
    7       .00      1.00    8       .00      1.00    9       .00      1.00
   10       .00      1.00   11       .00      1.00   12       .00      1.00
   13       .00      1.00   14       .00      1.00   15       .00      1.00
   16       .00      1.00   17       .00      1.00
```

```
    PO4                                 5  1.2   .10E+09
  1       .00      1.00    2       .00      1.00    3       .00      1.00
  4       .00      1.00    5       .00      1.00    6       .00      1.00
  7       .00      1.00    8       .00      1.00    9       .00      1.00
 10       .00      1.00   11       .00      1.00   12       .00      1.00
 13       .00      1.00   14       .00      1.00   15       .00      1.00
 16       .00      1.00   17       .00      1.00
   PHYT                                 4  1.2   .10E+09
  1       .00      1.00    2       .00      1.00    3       .00      1.00
  4       .00      1.00    5       .00      1.00    6       .00      1.00
  7       .00      1.00    8       .00      1.00    9       .00      1.00
 10       .00      1.00   11       .00      1.00   12       .00      1.00
 13       .00      1.00   14       .00      1.00   15       .00      1.00
 16       .00      1.00   17       .00      1.00
   CBOD                                 3  1.0   .10E+09
  1      7.18       .50    2      5.04       .50    3      5.04       .50
  4      5.04       .50    5      6.25       .50    6      6.25       .50
  7      5.06       .50    8      5.06       .50    9      5.06       .50
 10      8.14       .50   11      8.40       .50   12      8.40       .50
 13      8.00       .50   14      8.60       .50   15      8.60       .50
 16      6.00      1.00   17      7.42      1.00
     DO                                 5  1.0   .10E+09
  1      1.36      1.00    2      1.81      1.00    3      1.81      1.00
  4      1.81      1.00    5       .68      1.00    6       .68      1.00
  7      1.64      1.00    8      1.64      1.00    9      1.64      1.00
 10      3.03      1.00   11      5.00      1.00   12      5.00      1.00
 13      5.00      1.00   14      5.00      1.00   15      5.00      1.00
 16      5.00      1.00   17      3.63      1.00
     ON                                 3  1.2   .10E+09
  1       .00      1.00    2       .00      1.00    3       .00      1.00
  4       .00      1.00    5       .00      1.00    6       .00      1.00
  7       .00      1.00    8       .00      1.00    9       .00      1.00
 10       .00      1.00   11       .00      1.00   12       .00      1.00
 13       .00      1.00   14       .00      1.00   15       .00      1.00
 16       .00      1.00   17       .00      1.00
     OP                                 3  1.2   .10E+09
  1       .00      1.00    2       .00      1.00    3       .00      1.00
  4       .00      1.00    5       .00      1.00    6       .00      1.00
  7       .00      1.00    8       .00      1.00    9       .00      1.00
 10       .00      1.00   11       .00      1.00   12       .00      1.00
 13       .00      1.00   14       .00      1.00   15       .00      1.00
 16       .00      1.00   17       .00      1.00
```

Appendix E
Examples of Free Form Text Files Generated by Avenue

The following is an example of the text files written by Avenue from the "All Input Blocks" option under "BOD/DO Input Block" Menu

<u>a.txt -- created by script inputa</u>

```
0 17 8 0 0 4 1 0 0 0 0.05 100.0 0.5 8
```

<u>inptnme.txt -- created by script inputa</u>

```
aveyr.inp
```

<u>title.txt -- created by script inputa</u>

```
Average Year Model Run
```

<u>b.txt -- created by script inputb</u>

```
0 1 0 0 0
704.5 1625.8 3093.5 6 7
704.5 2471.2 2947.06 7 8
704.5 1625.8 3867.77 5 6
704.5 1625.8 3388.48 1 2
704.5 1625.8 3334.92 4 5
704.5 1625.8 2764.21 2 3
704.5 1625.8 2310.66 3 4
119.9 465.4 5642.69 10 1
119.9 168.2 1609.85 12 2
119.9 505.4 1609.85 11 3
119.9 185.8 1609.85 13 4
119.9 528.6 1609.85 14 5
119.9 717.2 1609.85 15 6
119.9 260.1 1609.85 16 8
704.5 2471.2 3311.43 17 8
959.3 2471.2 1609.35 17 0
0.001 3404.16 4.01 8 9
0.001 3697.06 3.1 6 9
0.001 2489.95 4.01 7 9
0.001 4038.48 4.62 5 9
0.001 3238.48 4.62 1 9
0.001 2631.37 3.1 4 9
0.001 3538.48 4.62 2 9
0.001 1989.95 4.62 3 9
555
```

<u>c.txt -- created by script inputc</u>

```
2 0 1.0 0
0.004 0 0 0
0.4 0 0 0
1.2 0 0 0
0.6 0 0 0
8 9 1 8.41237e+006
6 9 1 6.01067e+006
7 9 1 6.15316e+006
5 9 1 6.56576e+006
1 9 1 5.26512e+006
4 9 1 4.27808e+006
2 9 1 5.75286e+006
3 9 1 3.23526e+006
10 0 1 3.74503e+006
12 0 1 541385
11 0 1 1.62673e+006
13 0 1 598034
14 0 1 1.7014e+006
15 0 1 2.30845e+006
16 0 1 837184
17 0 1 7.95405e+006
9 0 3 250.28
```

<u>d.txt -- created by script inputd</u>

```
2 0
0 10
10 1
1 2
2 3
3 4
4 5
5 6
6 7
7 8
8 17
17 0
14.35 999
0 12
12 2
2 3
3 4
4 5
5 6
6 7
7 8
8 17
17 0
5.7 999
0 11
11 3
3 4
4 5
5 6
6 7
7 8
8 17
```

```
17 0
3.54 999
0 13
13 4
4 5
5 6
6 7
7 8
8 17
17 0
1 999
0 14
14 5
5 6
6 7
7 8
8 17
17 0
2.06 999
0 15
15 6
6 7
7 8
8 17
17 0
7.59 999
0 16
16 8
8 17
17 0
1.2 999
0 17
17 0
42.4 999
555 0
```

e.txt -- created by script inpute

```
8 0
10 1.92
12 1.37
11 1.87
13 1.68
14 1.67
15 1.48
16 1.35
17 7.42
10 3.03
12 5
11 5
13 5
14 5
15 5
16 5
17 3.63
```

f.txt -- created by script inputf

```
8 0
1 3.25666
2 24.1639
3 0
4 1827.28
5 70.0928
6 291.247
7 30.814
8 281.801
1 8007.16
2 3685.55
3 2080.65
4 647.397
5 1362.58
6 4427.35
7 116.044
8 692.011
```

g.txt -- created by script inputg

```
1.065 0 0 0
8 24 1.5 9.85
6 24.2 1.5 9.45
7 24 1.5 9.85
5 26.4 1.5 9.96
1 28 1.5 5.82
4 23.8 1.5 7.87
2 23.8 1.5 7.87
3 23.8 1.5 7.87
10 23.6 1.5 1.82
12 20 1.5 0.2
11 20 1.5 0.2
13 20 1.5 0.2
14 20 1.5 0.2
15 20 1.5 0.2
16 20 1.5 0.2
17 25.3 1.5 10.9
9 20 1.5 0
```

h.txt -- created by script inputh

```
0.1
0.1
```

i.txt -- created by script inputi

```
0
```

j.txt -- created by script inputj

```
1 0
1 0
```

```
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
1 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
1 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
1 0
1 0
2 0
```

```
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
0.5 0
1 7.18
2 5.04
3 5.04
4 5.04
5 6.25
6 3.53
7 5.06
8 5.06
9 0
10 8.14
11 8.4
12 6.9
13 8
14 8.6
15 5.9
16 6
17 7.42
1 0
1 1.36
2 1.81
3 1.81
4 1.81
5 0.68
6 2.25
7 1.64
8 1.64
9 0
10 3.03
11 5
12 5
13 5
14 5
15 5
16 5
17 3.63
1 0
1 0
2 0
3 0
```

```
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
1 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
1 0
1 0
```

```
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
1 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
```

Appendix F
Avenue Scripts Used for ArcView/WASP5 Connection

Scripts Created for ArcView/WASP5 Connection

Script Name	Function
all	Runs all scripts to create texts file for EUTRO5 input file
bttn	Controls the "bug" icon on the "Segmentation" view to plot a chart
calinputa	Writes the text files for Input Block A in the model calibration
calinputall	Runs scripts which create text files for TOXI5 input file
cal_parchk	Checks the parameters on the output file to process for the calibration model
eutrorun	Runs the EUTRO5 model from ArcView
frame1 - frame4	Produces a new theme on the "Segmentation" to observe the change in concentration over time; each scipt is time delayed
gen_file_eutro	Executes outgen.exe from ArcView
gen_file_toxi	Executes calgen.exe from ArcView
help	Executes the help file for the ArcView/WASP5 connection
inputa	Writes the text files for Input Block A for the BOD/DO model input file
inputb	Writes the text files for Input Block B for the BOD/DO or calibration model input file
inputc	Writes the text files for Input Block C for the BOD/DO or calibration model input file
inputcale	Writes the text files for Input Block E for the calibration input file
inputcalf	Writes the text files for Input Block F for the calibration input file
inputcalg	Writes the text files for Input Block G for the calibration input file
inputcalh	Writes the text files for Input Block H for the calibration input file
inputcalj	Writes the text files for Input Block J for the calibration input file
inputd	Writes the text files for Input Block D for the BOD/DO or calibration input file
inpute	Writes the text files for Input Block E for the BOD/DO input file
inputf	Writes the text files for Input Block F for the BOD/DO input file
inputg	Writes the text files for Input Block G for the BOD/DO input file
inputh	Writes the text files for Input Block H for the BOD/DO input file
inputi	Writes the text files for Input Block I for the BOD/DO or calibration input file
inputj	Writes the text files for Input Block J for the BOD/DO input file
mod_parchk	Checks the parameters on the output file to process for the BOD/DO model
run_calout	Executes calout.exe and creates a dbf table for salinity at each segment over time
run_modout	Executes modout.exe and creates dbf tables for DO and BOD at each segment over time
toxirun	Executes TOXI5 from ArcView
vwout	Controls the output presentation processes -- all five options

'Script: all

```
'this script runs all of the ten input blocks for
'the model run of eutro
theProject = av.GetProject

_dummy = 0
aSEd = av.GetProject.FindDoc("inputa")
aSEd.Run(_dummy)
bSEd = av.GetProject.FindDoc("inputb")
bSEd.Run(_dummy)
cSEd = av.GetProject.FindDoc("inputc")
cSEd.Run(_dummy)
dSEd = av.GetProject.FindDoc("inputd")
dSEd.Run(_dummy)
eSEd = av.GetProject.FindDoc("inpute")
eSEd.Run(_dummy)
fSEd = av.GetProject.FindDoc("inputf")
fSEd.Run(_dummy)
gSEd = av.GetProject.FindDoc("inputg")
gSEd.Run(_dummy)
hSEd = av.GetProject.FindDoc("inputh")
hSEd.Run(_dummy)
iSEd = av.GetProject.FindDoc("inputi")
iSEd.Run(_dummy)
jSEd = av.GetProject.FindDoc("inputj")
jSEd.Run(_dummy)
```

- -

'Script: bttn

```
'This script controls the 'bug' button that graphs a chart of salinity vs. Time for
'a chosen segment

theProject = av.GetProject

'setting the Tables and views
salTable = av.GetProject.FindDoc(_tblname)
salVTab = salTable.GetVTab
salFields = salVTab.GetFields
timeField = salVtab.FindField("time")
theView = av.GetProject.FindDoc("Segmentation")
segTheme = theView.FindTheme("Main Segmentation")
bcTheme = theView.FindTheme("Boundary Segmentation")
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab


'Getting the User to select a point

 theView = av.GetProject.FindDoc("Segmentation")
 p = theView.GetDisplay.ReturnUserPoint
 theThemes = theView.GetActiveThemes
 t = theThemes.get(0)
 theFTab = t.GetFTab
 if (t.CanSelect.Not) then
   exit
 end
 t.SelectByPoint(p,#VTAB_SELTYPE_NEW)
 recs=t.FindByPoint(p)
 if (recs.count = 0) then
  MsgBox.Info("No segment selected","")
  exit
 end
 for each rec in recs
   numrec = rec
 end
 gcField = theFTab.FindField("grid-code")
 seg = theFTab.ReturnValue(gcField,numrec)
 plotField = salFields.Get(seg)
 fldLst = {plotField}

' Creates a list of tables and allows the user to pick which one they
' want to view the output for -- these steps execute only if the user has not
' yet chosen a table.
if (timeField = nil) then
docList = theProject.GetDocs
tabList = List.Make
numdocs=docList.count
for each i in 0..(numdocs-1)
  dtype=(docList.get(i)).GetClass.GetClassName
  if (dtype="Table") then
    tabList.Add(docList.Get(i).GetName)
  end
end
_tblname = MsgBox.ChoiceAsString(tabList,"Choose the output table you want to work with","View Output")
salTable = av.GetProject.FindDoc(_tblname)
salVTab = salTable.GetVTab
salFields = salVTab.GetFields
timeField = salVtab.FindField("time")
end

'Allows the user to choose a color for the chart
colorlist=List.Make
colorlist.Add("blue")
colorlist.Add("yellow")
colorlist.Add("green")
colorlist.Add("red")
col=MsgBox.ChoiceAsString(colorlist,"Select a color for the chart","View Output")
if (col="blue") then
  _chcolor = Color.GetBlue
elseif (col="yellow") then
```

```
  _chcolor = Color.GetYellow
elseif (col="green") then
  _chcolor = Color.GetGreen
else (col="red")
  _chcolor = Color.GetRed
end

 step = salVTab.ReturnValue(timeField,1) - salVtab.ReturnValue(timeField,0)
'Makes the chart
   xChart = Chart.Make(salVTab,fldLst)
   xChart.SetSeriesFromRecords(false)
'   xChart.SetRecordLabelField(TimeField)
   xchartname=xchart.getname
   theProject.setActive(xchart)
   xChartDisp = xchart.GetChartDisplay
   xChartDisp.setType(#CHARTDISPLAY_line)
   xChartDisp.SetSeriesColor(0,_chcolor)
   the_x=xChart.GetXAxis
   the_y=xChart.GetYAxis
   the_x.SetTickLabelsVisible(false)
   the_x.SetMajorGridVisible(false)
   the_y.SetMajorGridVisible(True)
   the_x.SetCrossValue(0)
   the_y.SetCrossValue(0)
   the_x.SetLabelVisible(true)
   the_y.SetLabelVisible(true)
   xLegend=xChart.GetChartLegend
   xLegend.SetVisible(False)
   xChart.GetTitle.SetName("Segment "+seg.AsString)
   ylst = {"Salinity (ppt)","DO (mg/L)","BOD (mg/L)"}
   yname = MsgBox.ChoiceAsString(ylst,"Please enter the y-axis","View Output")
   the_x.SetName("Time -- Step = "++step.AsString++"days")
   the_y.SetName(yname)
   xchart.GetWin.Open
   keep = MsgBox.YesNo("Would you like to keep the chart?","ArcView",true)
   if (keep=false) then
     av.GetProject.removeDoc(xchart)
     exit
   else
     chname = MsgBox.Input("Name the Chart Window","View Output","")
     xchart.setname(chname)
   end
```

- -

'Script: calinputa

```
theProject = av.GetProject
_dir = MsgBox.Input("Enter the working directory.","Working Directory","c:\benaman\wasp")
_dir.asFileName.setCWD

'This program sets the initial model options.  Note, that presently, there are
'a few model options which are preset within the input file.  These include:
'1.  Backward differencing used in finite differencing solver
'2.  A transport file is always generated
'3.  The first 6 segments are those solutions which are printed to the screen while the model is running
'4.  The same maximum time step is used throughout the model.
'5.  The same print interval is used throughout the model.
'6.  The model parameters are all steady state at the present time.
'To change these settings, one must physcially go into the input file which is created in this interface
'and change the variables.  For more information, please consult the WASP5 User's Manual B.

'The following Tables are needed:
'  1.  Main Segmentation (segment arc attribute table)
'  2.  Main Segment Paramters
'  3.  Boundary Segments (Parameters of boundary segments)

tle = MsgBox.Input("Please write a title for the model run (No more than 60 charaters)","Input File A: Model
Options","Test Run")

'Choosing the tables that are important
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
parTable = av.GetProject.FindDoc("Main Segment Parameters")
parVTab = parTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

'Joining the main segmentation table with its parameters

segFields = segVTab.GetFields
parFields = parVTab.GetFields
jtofield = segFields.Get(5)
jfromfield = parFields.Get(0)
segVTab.Join(jtofield,parVTab,jfromfield)

'Determining the number of segments (main and boundary)
_seg = (segVTab.GetNumRecords)+(bcVTab.GetNumRecords)

'Asking user for other prefernces dealiing with WASP file
icfl = MsgBox.Input("Do you want the model to read or write to a restart file?","Input File A: Model Options","N")
  if (icfl = "N") then
      icfl = 0 else
      icfl = 1
  end

mflag = MsgBox.Input("Do you want all error messages printed to the screen?","Input File A: Model Options","Y")
  if (mflag = "Y") then
    mflag = 0 else
    mflag = 1
  end
```

```
negsln = MsgBox.Input("Do you want to prevent negative solutions?","Input File A: Model Options","Y")
  if (negsln = "Y") then
    negsln = 0 else
    negsln = 1
  end

zlst = {"Day","Hour","Minute"}
zdef = {"0","0","0"}
z = MsgBox.MultiInput("Please enter the start time.","Input File A: Model Options",zlst,zdef)

_dts = MsgBox.Input("What is the maximum time step allowed (/day)?","Input File A: Model Options","0.001")

_tend = MsgBox.Input("How many days would you like to run the model?","Input File A: Model Options","100.0")

_prn = MsgBox.Input("At what interval (in days) would you like the results printed to the output file?","Input File A:
Model Options","1.0")

mnsegs = segVTab.GetNumRecords

fname=MsgBox.Input("Please enter the filename for the model input file","Input File A: Model Options","*.inp")

flnametxt = LineFile.Make("inptnme.txt".AsFileName,#FILE_PERM_WRITE)
flnametxt.WriteElt(fname)
titlefile = LineFile.Make("title.txt".AsFileName,#FILE_PERM_WRITE)
titlefile.WriteElt(tle)
atxt = LineFile.Make("cala.txt".AsFileName,#FILE_PERM_WRITE)
otpta =  _seg.AsString++icfl.asString++mflag.asString
otpta = otpta++negsln.asString

for each i in z
otpta = otpta++i.AsString
end
otpta = otpta++_dts.AsString++_tend.AsString++_prn.AsString++mnsegs.AsString
atxt.WriteElt(otpta)

MsgBox.Info("Done Writing Input Block A","")
```

- -

'Script:calinputall

```
'This script runs all ten input block scripts for the calibration input

theProject = av.GetProject

_dummy = 0
aSEd = av.GetProject.FindDoc("calinputa")
aSEd.Run(_dummy)
bSEd = av.GetProject.FindDoc("inputb")
bSEd.Run(_dummy)
cSEd = av.GetProject.FindDoc("inputc")
cSEd.Run(_dummy)
dSEd = av.GetProject.FindDoc("inputd")
dSEd.Run(_dummy)
eSEd = av.GetProject.FindDoc("inputcale")
eSEd.Run(_dummy)
fSEd = av.GetProject.FindDoc("inputcalf")
fSEd.Run(_dummy)
gSEd = av.GetProject.FindDoc("inputcalg")
gSEd.Run(_dummy)
hSEd = av.GetProject.FindDoc("inputcalh")
hSEd.Run(_dummy)
iSEd = av.GetProject.FindDoc("inputi")
iSEd.Run(_dummy)
jSEd = av.GetProject.FindDoc("inputcalj")
jSEd.Run(_dummy)
```

- -

'Script: cal_parchk

```
'This script checks the parameters of the calibration output file, before you
'run the processing output step

theProject = av.GetProject

_dir = MsgBox.Input("Enter the working directory.","Working Directory","c:\benaman\wasp")
_dir.AsFilename.setCWD
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab
starttxt = LineFile.Make("start.txt".AsFileName,#FILE_PERM_WRITE)

_seg = (segVTab.GetNumRecords)+(bcVTab.GetNumRecords)

_dts = MsgBox.Input("What was the maximum time step allowed (/day)?","Input File A: Model Options","0.001")

_tend = MsgBox.Input("How many days did you run the model?","Input File A: Model Options","100.0")

_prn = MsgBox.Input("At what interval (in days) did you have the results printed to the output file?","Input File A:
Model Options","1.0")

outname = MsgBox.Input("Enter the name of the file you want to process","Calibration Output Processing","*.tdf")
outtxt = LineFile.Make("outnme.txt".AsFileName,#FILE_PERM_WRITE)
outtxt.WriteElt(outname)

qu = MsgBox.YesNo("Is the output file called"++outname+"?","Calibration Output Processing",TRUE)
if (qu=TRUE) then
lst = "Is the following correct?"+nl+"Number of Days Run:"++_tend.AsString++"days"+nl+"Print Interval for output
file:"++_prn.AsString++"days"+nl+"Number of Segments:"++_seg.AsString
qu2 = MsgBox.YesNo(lst,"Calibration Output Processing",TRUE)
  if (qu2=TRUE) then
  starttxt.WriteElt(_tend.AsString++_dts.AsString++_prn.AsString++_seg.AsString)
  else
  MsgBox.Info("Please see the Help for how to set up the output for processing","")
  end
else
MsgBox.Info("Please see the Help for how to set up the output for processing","")
exit
end
```

- -

'Script: eutrorun

```
'This script executes the WASP model EUTRO5
'it allows you to choose any input file, but the one
'generated by this connection will be called "test.inp"

_dir.asFileName.setCWD
system.execute("eutro5.exe")
MsgBox.Info("Click OK when EUTRO5 is done running","")
```

- -

'Script: frame1

```
'This script produces the first frame of the "movie"
'It is called by, and dependent on, the script, vwout

theProject = av.GetProject

'getting parameters passed by the main script
segSrcName = SELF.Get(0)
concname = SELF.Get(1)
time = SELF.Get(2)
theView = SELF.Get(3)
movVTab = SELF.Get(4)
x = SELF.Get(5)
n = SELF.Get(6)
inv = SELF.Get(7)

'Setting the intervals for the classifications of the legend

x1 = n
x2 = inv + n
x3 = 2*inv + n
x4 = 3*inv + n
x5 = 4*inv + n
x6 = 5*inv + n
x7 = 6*inv + n
x8 = 7*inv + n
x9 = x

'Making the theme
newTheme = Theme.Make(segSrcName)
newTheme.SetVisible(true)
theView.AddTheme(newTheme)
theView.Invalidate
newFTab = newTheme.GetFTab
movFields = movVTab.GetFields

'Gets the FTab for the new theme and joins the newly made
'temp.dbf to the aat of the theme

   newFields = newFTab.GetFields
   jtoField = newFields.Get(8)
   jfromField = movFields.Get(0)
   newFTab.Join(jtoField,movVTab,jfromField)

'setting the classifications of the legend
   a = Classification.Make(x1,x2)
   b = Classification.Make(x2,x3)
   c = Classification.Make(x3,x4)
   d = Classification.Make(x4,x5)
   e = Classification.Make(x5,x6)
   f = Classification.Make(x6,x7)
   g = Classification.Make(x7,x8)
   h = Classification.Make(x8,x9)

   ClassLst = {a,b,c,d,e,f,g,h}
     newTheme.SetName(concname.AsString++" at "++time.AsString)
     lookField = newFTab.FindField("conc1")
     theLegend = newTheme.GetLegend
     theLegend.Quantile(newFTab,lookField,8)
     theClassList = theLegend.GetClassifications
     cnt = 0
     for each i in theClassList
       theClassList.Set(cnt,ClassLst.Get(cnt))
       cnt = cnt + 1
     end
     newTheme.UpdateLegend
'ramping the colors and making it a larger line
     theLegend.RampColors(Color.GetGray,Color.GetBlue)
     c = 0
     for each i in theLegend.GetSymbols
     if (c < 4) then
       i.SetWidth(2)
     else
       i.SetWidth(3)
     end
     c = c+1
     end
     newTheme.UpdateLegend
     theView.InValidate
```

- -

'Script: frame2

```
'This script produces the second frame of the "movie"
'It is called by, and dependent on, the script, vwout

theProject = av.GetProject

'getting parameters passed by the main script
segSrcName = SELF.Get(0)
concname = SELF.Get(1)
time = SELF.Get(2)
theView = SELF.Get(3)
movVTab = SELF.Get(4)
x = SELF.Get(5)
n = SELF.Get(6)
inv = SELF.Get(7)

'Setting the intervals for the classifications of the legend

x1 = n
x2 = inv + n
x3 = 2*inv + n
x4 = 3*inv + n
x5 = 4*inv + n
x6 = 5*inv + n
x7 = 6*inv + n
x8 = 7*inv + n
x9 = x

'Making the theme
newTheme = Theme.Make(segSrcName)
newTheme.SetVisible(true)
theView.AddTheme(newTheme)
theView.Invalidate
newFTab = newTheme.GetFTab
movFields = movVTab.GetFields

'Gets the FTab for the new theme and joins the newly made
'temp.dbf to the aat of the theme

   newFields = newFTab.GetFields
   jtoField = newFields.Get(8)
   jfromField = movFields.Get(0)
   newFTab.Join(jtoField,movVTab,jfromField)

'setting the classifications of the legend
   a = Classification.Make(x1,x2)
   b = Classification.Make(x2,x3)
   c = Classification.Make(x3,x4)
   d = Classification.Make(x4,x5)
   e = Classification.Make(x5,x6)
   f = Classification.Make(x6,x7)
   g = Classification.Make(x7,x8)
   h = Classification.Make(x8,x9)

   ClassLst = {a,b,c,d,e,f,g,h}
     newTheme.SetName(concname.AsString++" at "++time.AsString)
     lookField = newFTab.FindField("conc2")
     theLegend = newTheme.GetLegend
     theLegend.Quantile(newFTab,lookField,8)
     theClassList = theLegend.GetClassifications
     cnt = 0
     for each i in theClassList
       theClassList.Set(cnt,ClassLst.Get(cnt))
       cnt = cnt + 1
     end
     newTheme.UpdateLegend
'ramping the colors and making it a larger line
    theLegend.RampColors(Color.GetGray,Color.GetBlue)
     c = 0
     for each i in theLegend.GetSymbols
     if (c < 4) then
       i.SetWidth(2)
     else
       i.SetWidth(3)
     end
     c = c+1
     end
     newTheme.UpdateLegend
     theView.InValidate
```

- -

'Script: frame3

```
'This script produces the third frame of the "movie"
'It is called by, and dependent on, the script, vwout

theProject = av.GetProject

'getting parameters passed by the main script
segSrcName = SELF.Get(0)
concname = SELF.Get(1)
time = SELF.Get(2)
theView = SELF.Get(3)
movVTab = SELF.Get(4)
x = SELF.Get(5)
n = SELF.Get(6)
inv = SELF.Get(7)

'Setting the intervals for the classifications of the legend
```

```
x1 = n
x2 = inv + n
x3 = 2*inv + n
x4 = 3*inv + n
x5 = 4*inv + n
x6 = 5*inv + n
x7 = 6*inv + n
x8 = 7*inv + n
x9 = x

'Making the theme
newTheme = Theme.Make(segSrcName)
newTheme.SetVisible(true)
theView.AddTheme(newTheme)
theView.Invalidate
newFTab = newTheme.GetFTab
movFields = movVTab.GetFields

'Gets the FTab for the new theme and joins the newly made
'temp.dbf to the aat of the theme

   newFields = newFTab.GetFields
   jtoField = newFields.Get(8)
   jfromField = movFields.Get(0)
   newFTab.Join(jtoField,movVTab,jfromField)

'setting the classifications of the legend
   a = Classification.Make(x1,x2)
   b = Classification.Make(x2,x3)
   c = Classification.Make(x3,x4)
   d = Classification.Make(x4,x5)
   e = Classification.Make(x5,x6)
   f = Classification.Make(x6,x7)
   g = Classification.Make(x7,x8)
   h = Classification.Make(x8,x9)

   ClassLst = {a,b,c,d,e,f,g,h}
     newTheme.SetName(concname.AsString++" at "++time.AsString)
     lookField = newFTab.FindField("conc3")
     theLegend = newTheme.GetLegend
     theLegend.Quantile(newFTab,lookField,8)
     theClassList = theLegend.GetClassifications
     cnt = 0
     for each i in theClassList
       theClassList.Set(cnt,ClassLst.Get(cnt))
       cnt = cnt + 1
     end
     newTheme.UpdateLegend
'ramping the colors and making it a larger line
      theLegend.RampColors(Color.GetGray,Color.GetBlue)
     c = 0
     for each i in theLegend.GetSymbols
     if (c < 4) then
       i.SetWidth(2)
     else
       i.SetWidth(3)
     end
     c = c+1
     end
     newTheme.UpdateLegend
     theView.InValidate
```

- -

'Script: frame4

```
'This script produces the fourth frame of the "movie"
'It is called by, and dependent on, the script, vwout

theProject = av.GetProject

'getting parameters passed by the main script
segSrcName = SELF.Get(0)
concname = SELF.Get(1)
time = SELF.Get(2)
theView = SELF.Get(3)
movVTab = SELF.Get(4)
x = SELF.Get(5)
n = SELF.Get(6)
inv = SELF.Get(7)

'Setting the intervals for the classifications of the legend

x1 = n
x2 = inv + n
x3 = 2*inv + n
x4 = 3*inv + n
x5 = 4*inv + n
x6 = 5*inv + n
x7 = 6*inv + n
x8 = 7*inv + n
x9 = x

'Making the theme
newTheme = Theme.Make(segSrcName)
newTheme.SetVisible(true)
theView.AddTheme(newTheme)
theView.Invalidate
newFTab = newTheme.GetFTab
movFields = movVTab.GetFields

'Gets the FTab for the new theme and joins the newly made
```

```
'temp.dbf to the aat of the theme

  newFields = newFTab.GetFields
  jtoField = newFields.Get(8)
  jfromField = movFields.Get(0)
  newFTab.Join(jtoField,movVTab,jfromField)

'setting the classifications of the legend
  a = Classification.Make(x1,x2)
  b = Classification.Make(x2,x3)
  c = Classification.Make(x3,x4)
  d = Classification.Make(x4,x5)
  e = Classification.Make(x5,x6)
  f = Classification.Make(x6,x7)
  g = Classification.Make(x7,x8)
  h = Classification.Make(x8,x9)

  ClassLst = {a,b,c,d,e,f,g,h}
   newTheme.SetName(concname.AsString++" at "++time.AsString)
   lookField = newFTab.FindField("conc4")
   theLegend = newTheme.GetLegend
   theLegend.Quantile(newFTab,lookField,8)
   theClassList = theLegend.GetClassifications
   cnt = 0
   for each i in theClassList
     theClassList.Set(cnt,ClassLst.Get(cnt))
     cnt = cnt + 1
   end
   newTheme.UpdateLegend
'ramping the colors and making it a larger line
   theLegend.RampColors(Color.GetGray,Color.GetBlue)
   c = 0
   for each i in theLegend.GetSymbols
   if (c < 4) then
     i.SetWidth(2)
   else
     i.SetWidth(3)
   end
   c = c+1
   end
   newTheme.UpdateLegend
   theView.InValidate
```

- -

'Script: gen_file_eutro

```
'This script executes the FORTRAN program which takes all 12
'of the text files created from Scripts inputa through
'inputj and formats them into one large input file for EUTRO
'The input file is ALWAYS called test.inp
_dir.asFileName.setCWD
system.execute("outgen.exe")
MsgBox.Info("Done generating WASP5 input file","")
```

- -

'Script: gen_file_toxi

```
theProject = av.GetActiveDoc
'This script executes the FORTRAN program which takes all 12
'of the text files created from Scripts inputa through
'inputj and formats them into one large input file for TOXI
'for the model calibration

_dir.asFileName.setCWD
system.execute("calgen.exe")
MsgBox.Info("Done generating WASP5 input file","")
```

- -

'Script: help

```
'this script executes the help file, written for winhelp.exe

system.execute("winhelp.exe c:\benaman\winhelp\wasptm.hlp")
```

- -

'Script: inputa

```
theProject = av.GetProject
_dir = MsgBox.Input("Enter the working directory.","Working Directory","c:\benaman\wasp")
_dir.asFileName.setCWD


'This program sets the initial model options.  Note, that presently, there are
'a few model options which are preset within the input file.  These include:
'1.  Backward differencing used in finite differencing solver
'2.  A transport file is always generated
'3.  The first 6 segments are those solutions which are printed to the screen while the model is running
'4.  The same maximum time step is used throughout the model.
'5.  The same print interval is used throughout the model.
'6.  The model parameters are all steady state at the present time.
'To change these settings, one must physcially go into the input file which is created in this interface
'and change the variables.  For more information, please consult the WASP5 User's Manual B.

'The following Tables are needed:
'  1.  Main Segmentation (segment arc attribute table)
'  2.  Main Segment Paramters
'  3.  Boundary Segments (Parameters of boundary segments)

mdlList = {"Simple Streeter-Phelps with SOD","Modified Streeter Phelps with NBOD","Linear DO Balance with
Nitrification","Nonlinear DO Balance","Simple Eutrophication","Intermediate Eutrophication","Intermediate
Eutrophication with Benthos"}
for each i in 1..6
mdl = MsgBox.ChoiceAsString(mdlList,"What type of model would you like to run?","Input File A:  Model Options")
```

```
_imdl=mdlList.Find(mdl)
  if (_imdl > 0) then
    MsgBox.Info("Unable to run that Level at this time.","Sorry")
    else
    break
  end
end
tle = MsgBox.Input("Please write a title for the model run (No more than 60 charaters)","Input File A: Model
Options","Test Run")

'Choosing the tables that are important
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
parTable = av.GetProject.FindDoc("Main Segment Parameters")
parVTab = parTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

'Joining the main segmentation table with its parameters

segFields = segVTab.GetFields
parFields = parVTab.GetFields
jtofield = segFields.Get(5)
jfromfield = parFields.Get(0)
segVTab.Join(jtofield,parVTab,jfromfield)

'Determining the number of segments (main and boundary)
seg = (segVTab.GetNumRecords)+(bcVTab.GetNumRecords)

'Asking for the number of systems (i.e. constiuents to be modeled -- default for EUTRO is 8)
for each i in 1..3
sys = MsgBox.Input("Please enter the number of systems.","Input File A: Model Options","8")
  sys = sys.AsNumber
  if (sys = 8) then
   break
  else
    MsgBox.Info("EUTRO5 always requires 8 systems","")
    continue
  end
end
'Asking user for other prefernces dealiing with WASP file
icfl = MsgBox.Input("Do you want the model to read or write to a restart file?","Input File A: Model Options","N")
  if (icfl = "N") then
      icfl = 0 else
      icfl = 1
  end

mflag = MsgBox.Input("Do you want all error messages printed to the screen?","Input File A: Model Options","Y")
  if (mflag = "Y") then
    mflag = 0 else
    mflag = 1
  end

massList = {"NH3","NO3","PO4","Chla","CBOD","DO","ON","OP"}
massys = MsgBox.ChoiceAsString(massList,"Choose the system for which the mass balance will be performed","Input
File A: Model Options")
imassys = massList.Find(massys)

negsln = MsgBox.Input("Do you want to prevent negative solutions?","Input File A: Model Options","Y")
  if (negsln = "Y") then
    negsln = 0 else
    negsln = 1
  end

zlst = {"Day","Hour","Minute"}
zdef = {"0","0","0"}
z = MsgBox.MultiInput("Please enter the start time.","Input File A: Model Options",zlst,zdef)

_dts = MsgBox.Input("What is the maximum time step allowed (/day)?","Input File A: Model Options","0.001")

_tend = MsgBox.Input("How many days would you like to run the model?","Input File A: Model Options","100.0")

_prn = MsgBox.Input("At what interval (in days) would you like the results printed to the output file?","Input File A:
Model Options","1.0")

mnsegs = segVTab.GetNumRecords
fname=MsgBox.Input("Please enter the filename for the model input file","Input File A: Model Options","*.inp")

flnametxt = LineFile.Make("inptnme.txt".AsFileName,#FILE_PERM_WRITE)
flnametxt.WriteElt(fname)
titlefile = LineFile.Make("title.txt".AsFileName,#FILE_PERM_WRITE)
titlefile.WriteElt(tle)
atxt = LineFile.Make("a.txt".AsFileName,#FILE_PERM_WRITE)
otpta =  _imdl.AsString++seg.AsString++sys.AsString++icfl.asString++mflag.asString++imassys.asString
otpta = otpta++negsln.asString

for each i in z
otpta = otpta++i.AsString
end
otpta = otpta++_dts.AsString++_tend.AsString++_prn.AsString++mnsegs.AsString
atxt.WriteElt(otpta)

MsgBox.Info("Done Writing Input Block A","")
```

. . . .

'Script: inputb

```
'This script generates the free format text file for Input Block B
'The following defaults are set:
'  1.  All exchanges are steady state
'Tables needed: Same as inputa
```

```
theProject = av.GetProject
_dir.asFileName.setCWD
'_imdl = the type of model being run

btxt = LineFile.Make("b.txt".AsFileName,#FILE_PERM_WRITE)

'get the initial model information

exList = {"Water Column Only","Water Column and Pore Water"}
nrfld = MsgBox.ChoiceAsString(exList,"What fields undergo exchange?","Input File B: Exchange Coefficients")
inrfld=exList.Find(nrfld)
  if (inrfld = 0) then
    nrfld = 1 else
    nrfld = 2
  end
if (_imdl=nil) then
_imdl = 0
end

lst = _imdl.AsString++nrfld.AsString++"0"++"0"++"0"
btxt.WriteElt(lst)

'identify the two tables that have the segment information in them

segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

'Now concentrate on the main segments first: Get the exchange coefficients,
'characteristic length, and neighboring segment number

srt1Field = segTable.GetVTab.FindField("Grid-Code")
if (segTable.GetWin.IsOpen) then
  segTable.Sort( srt1Field, FALSE )
end

gcField = segVTab.FindField("grid-code")
dwnField = segVTab.FindField("dwnstr_seg")
araField = segVTab.FindField("x_area")
exField = segVTab.FindField("ex_coeff")
lnField = segVTab.FindField("length")

For Each rec in segVTab
  seg = segVTab.ReturnValue(gcField,rec)
  dwnstr = segVTab.ReturnValue(dwnField,rec)
  area1 = segVTab.ReturnValue(araField,rec)
  ex1 = segVTab.ReturnValue(exField,rec)
  length1=segVTab.ReturnValue(lnField,rec)

'Finding the same values for the downstream segment

    For Each reca in segVTab
      dwnseg = segVTab.ReturnValue(gcField,reca)
     if (dwnseg = dwnstr) then
        tmprec = reca.AsString
        itmprec = tmprec.AsNumber
        break
        else
        tmprec = "999"
        itmprec = tmprec.AsNumber
      end
    end

     if (itmprec = 999) then
     MsgBox.Info("The segment downstream of segment"++seg.AsString++"is a boundary segment.","")
     else
     area2 = segVTab.ReturnValue(araField,itmprec)
     ex2 = segVTab.ReturnValue(exField,itmprec)
     length2=segVTab.ReturnValue(lnField,itmprec)

'comparison of the sgement values and its downstream segment values
       if (area1 > area2) then
        area = area2
        else
        area = area1
       end
       if (ex1 > ex2) then
         ex = ex1
         else
         ex = ex2
       end
       chlth = (length1 + length2)/2
'writing the information to the file
       exclst = ex.AsString++area.AsString++chlth.AsString++seg.AsString++dwnstr.AsString
       btxt.WriteElt(exclst)
     end
  end

'Now concentrate on the boundary segments

bcVTab = bcTable.GetVTab
perField = bcVTab.FindField("perpin")
bcgcField = bcVTab.FindField("grid-code")
bcdwnField = bcVTab.FindField("dwnstr_seg")
upField = bcVTab.FindField("upstr_seg")
bcaraField = bcVTab.FindField("x_area")
bcexField = bcVTab.FindField("ex_coeff")
bclnField = bcVTab.FindField("act_length")
```

```
wdField = segVTab.FindField("width")
typField = bcVTab.FindField("type")

srt1Field = bcTable.GetVTab.FindField("Grid-Code")
if (bcTable.GetWin.IsOpen) then
  bcTable.Sort( srt1Field, FALSE )
end

For Each rec in bcVTab
  seg = bcVTab.ReturnValue(bcgcField,rec)
  dwnstr = bcVTab.ReturnValue(bcdwnField,rec)
  type = bcVTab.ReturnValue(typField,rec)
  if (type <> 1) then
    break
  end
    if (dwnstr = 0) then
 'allows the most downstream segment to look at its upstream segment
      dwnstr = bcVTab.ReturnValue(upField,rec)
      'assumes the most downstream segment has a dispersive boundary with 0
      lastseg = seg
      lastex = bcVTab.ReturnValue(bcexField,rec)
      lastarea = bcVTab.ReturnValue(bcaraField,rec)
      lastchlth = (bcVTab.ReturnValue(bclnField,rec)/2)
    end
  area1 = bcVTab.ReturnValue(bcaraField,rec)
  ex1 = bcVTab.ReturnValue(bcexField,rec)
  length1=bcVTab.ReturnValue(bclnField,rec)
  pi = bcVTab.ReturnValue(perField,rec)
  p = pi.AsString

'Finding the same values for the downstream segment

    For Each reca in segVTab
      dwnseg = segVTab.ReturnValue(gcField,reca)
     if (dwnseg = dwnstr) then
        tmprec = reca.AsString
        itmprec = tmprec.AsNumber
        break
        else
        tmprec = "999"
        itmprec = tmprec.AsNumber
      end
    end

     if (itmprec = 999) then
      dummy = 0
     else
     area2 = segVTab.ReturnValue(araField,itmprec)
     length2=segVTab.ReturnValue(lnField,itmprec)
     wdth = segVTab.ReturnValue(wdField,itmprec)
     ex2 = segVTab.ReturnValue(exField,itmprec)
     end

'comparison of the segement values and its downstream segment values
       if (area1 > area2) then
        area = area2
        else
        area = area1
       end

'checks to see if the boundary is perpindicular
       if (p = "N")
       then
       chlth = (length1 + length2)/2
         if (ex1 > ex2) then
           ex = ex2
         else
           ex = ex1
          end
       else
       chlth = (wdth + length1)/2
       area = area1
       ex = ex1
       end

'writing the information to the file
       if (ex=0) then
       break
       end
       bcexclst = ex.AsString++area.AsString++chlth.AsString++seg.AsString++dwnstr.AsString
       btxt.WriteElt(bcexclst)

  end
     bcexlst = lastex.AsString++lastarea.AsString++lastchlth.AsString++lastseg.AsString++"0"
     btxt.WriteElt(bcexlst)

 'Now look at the exchange between the sediment layer and the overlying water body

 bcdpField = bcVTab.FindField("depth")
 'Find the record the sediment layer is in
 For each rec in bcVTab
   type = bcVTab.ReturnValue(typField,rec)
   if (type = 3) then
     sedex = bcVTab.ReturnValue(bcexField,rec)
     sedgc = bcVTab.ReturnValue(bcgcField,rec)
     seddth = bcVTab.ReturnValue(bcdpField,rec)
   end
 end

 dpField = segVTab.FindField("depth")
```

```
For each rec in segVTab
  area = (segVTab.ReturnValue(lnField,rec))*(segVTab.ReturnValue(wdField,rec))
  dp = segVTab.ReturnValue(dpField,rec)
  chln = (dp+seddth)/2
  seg = segVTab.ReturnValue(gcField,rec)
  bcexlst = sedex.AsString++area.AsString++chln.AsString++seg.AsString++sedgc.AsString
  btxt.WriteElt(bcexlst)
end

 btxt.WriteElt("555")
  MsgBox.Info("Done writing Input Block B","")
```

- -

'Script: inputc

```
'This script writes the free form text file used to create
'Input Block C
'The following defaults are set in this file:
'  1.  The hydraulic coefficients used, in WASP, to calculate volitilization
'      and reaeration do not spatially vary
'Tables Needed: Same as inputa

theProject = av.GetProject
_dir.asFileName.setCWD

ctxt = LineFile.Make("c.txt".AsFileName,#FILE_PERM_WRITE)

'get the initial model information

volList = {"Constant Water Column Volume","Volumes Adjusted to Maintain Flow Continuity"}
vopt= MsgBox.ChoiceAsString(volList,"Choose a water column volume option.","Input File C: Volumes")
ivopt=(volList.Find(vopt))+1

for each i in 1..2
bedList = {"Constant Bed Volume","Volumes Adjusted to Respond to Sediment Transport"}
bedv= MsgBox.ChoiceAsString(bedList,"Choose a benthic volume option","Input File C: Volumes")
ibedv=(bedList.Find(bedv))
if (ibedv <> 0) then
  MsgBox.Info("Sediment transport is currently not simulated in this model","Input File C: Volumes")
  else
  break
end
end

tdints = MsgBox.Input("Please enter the benthic time step for recomputing porosity (/day).","Input File
C:Volumes","")

lst = ivopt.AsString++ibedv.AsString++tdints.AsString++"0"
ctxt.WriteElt(lst)

'get the hydraulic coefficients

hydlist = {"a","b","c","d"}
hyddef = {"0.004","0.4","1.2","0.6"}

hydcoef = MsgBox.MultiInput("Enter the coef. for the eqns: v=aQ^b and d=Q^c","Input File
C:Volumes",hydlist,hyddef)

for each i in hydcoef
  ctxt.WriteElt(i.AsString++"0"++"0"++"0")
end

'identify the two tables that have the segment information in them

segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

gcField = segVTab.FindField("grid-code")
lnField = segVTab.FindField("length")
araField = segVTab.FindField("x_area")
botField = segVTab.FindField("bttm_seg")
typField = segVTab.FindField("type")

'Get the volume, segment type, and bottom segment for each segment
'First the main segments, then the boundary segments

for each rec in segVTab
  seg = segVTab.ReturnValue(gcField,rec)
  len = segVTab.ReturnValue(lnField,rec)
  area = segVTab.ReturnValue(araField,rec)
  botseg = segVTab.ReturnValue(botField,rec)
  type = segVTab.ReturnValue(typField,rec)

  volume = len * area

  lst = seg.AsString++botseg.AsString++type.AsString++volume.AsString
  ctxt.WriteElt(lst)
end

gcField = bcVTab.FindField("grid-code")
lnField = bcVTab.FindField("act_length")
araField = bcVTab.FindField("x_area")
botField = bcVTab.FindField("bttm_seg")
typField = bcVTab.FindField("type")

for each rec in bcVTab
  seg = bcVTab.ReturnValue(gcField,rec)
  len = bcVTab.ReturnValue(lnField,rec)
  area = bcVTab.ReturnValue(araField,rec)
  botseg = bcVTab.ReturnValue(botField,rec)
  type = bcVTab.ReturnValue(typField,rec)

  volume = len * area

  lst = seg.AsString++botseg.AsString++type.AsString++volume.AsString
  ctxt.WriteElt(lst)
end

MsgBox.Info("Done writing Input Block C","")
```

- -

'Script: inputcale

```
'This script writes the input block e for
'the calibration file
'Tables needed:
'1.Boundary Segments

theProject = av.GetProject
_dir.asFileName.setCWD

etxt = LineFile.Make("cale.txt".AsFileName,#FILE_PERM_WRITE)
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

numrecs = bcVTab.GetNumRecords
typField = bcVTab.FindField("type")
gcField = bcVTab.FindField("grid-code")
salField = bcvTab.FindField("sal")

'Find the number of water segments
for each rec in bcVTab
  type = bcVTab.ReturnValue(typField, rec)
  if (type = 3) then
   numrecs = numrecs - 1
  end
end

etxt.WriteElt(numrecs.asString++"0")

lst = list.Make
'get the salinity conc for each boundary
nobc = MsgBox.Input("How many boundaries are you going to set","Input File E: Boundaries","2")
nobc = nobc.AsNumber
For Each i in 1..nobc
  x =MsgBox.Input("Enter the segment number","Input File J: Initial Conditions","")
'obtain the salinity initial conditions and write them
    for each rec in bcVTab
      seg = bcVTab.ReturnValue(gcField,rec)
      sal = bcVTab.ReturnValue(salField,rec)
       if (seg = x.AsNumber) then
        lst.Add(rec)
        etxt.WriteElt(seg.asString++sal.asString)
        break
        else
        continue
        end
     end
end
temp = 555
for each rec in bcVTab
  seg = bcVTab.ReturnValue(gcField,rec)
  sal = bcVTab.ReturnValue(salField,rec)
  type = bcVtab.ReturnValue(typField,rec)
  if (type = 3)
  then
  break
  end
    for each i in lst
     if (i = rec) then
       temp = 999
       break
       else
       temp = 555
     end
    end
  if (temp = 999) then
  continue
  else
  etxt.WriteElt(seg.asString++"0.2")
  end
end
```

- -

'Script: inputcalf

```
'This script writes the input block f, for
'the calibration file
'Presently, there are no salinity loads into the
'system

theProject = av.GetProject
_dir.asFileName.setCWD

ftxt = LineFile.Make("calf.txt".AsFileName,#FILE_PERM_WRITE)
ftxt.WriteElt("0")
MsgBox.Info("There are no loads of salinity for the calibration input file.","Input File F: Loads")
```

'Script:inputcalg

```
'this script writes the input block g for the
'calibration file, presently, there are no
'parameters needed for the level one complexity
'of TOXI5

theProject = av.GetProject
_dir.asFileName.setCWD

gtxt = LineFile.Make("calg.txt".AsFileName,#FILE_PERM_WRITE)
gtxt.WriteElt("0")
MsgBox.Info("There are no parameters need for Level One Calibration","Input File G: Parameters")
```

- -

'Script: inputcalh

```
'This script writes the input block h for the
'calibration output file
'Tables needed: NONE

theProject = av.GetProject
_dir.asFileName.setCWD

htxt = LineFile.Make("calh.txt".AsFileName,#FILE_PERM_WRITE)

'Get the constants Kd and Ka from the user and write them to a file

lst = {"Solids-Independent Partition Coefficient(L/kg)","Water Column Biodegradation","Molecular Weight(g/mol)"}
lstdef = {"0.0","0.0","78.5"}
cnst = MsgBox.MultiInput("Please Enter the Following Constants","Input File H: Constants",lst,lstdef)

for each i in cnst
  htxt.WriteElt(i.AsString)
end

MsgBox.Info("Done writing Input Block H","")
```

- -

'Script: inputcalj

```
'This script writes the input block j
'for the calibration input file
'Tables Needed:
'1.Main Segmentation
'2.Boundary Segmentation

theProject = av.GetProject
_dir.asFileName.setCWD

jtxt = LineFile.Make("calj.txt".AsFileName,#FILE_PERM_WRITE)

'get the necessary tables
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

bcgcField = bcVTab.FindField("grid-code")
bcsalField = bcVTab.FindField("sal")

nobc = MsgBox.Input("How many boundaries are you going to set","Input File J: Initial Conditions","2")
nobc = nobc.AsNumber
jtxt.WriteElt("1.0"++"0") 'writing the dissolved fraction
jtxt.WriteElt(nobc.AsString++"0") 'writing the number of boundary conditions
For Each i in 1..nobc
  x =MsgBox.Input("Enter the segment number","Input File J: Initial Conditions","")
'obtain the salinity initial conditions and write them
    for each rec in bcVTab
      seg = bcVTab.ReturnValue(bcgcField,rec)
      sal = bcVTab.ReturnValue(bcsalField,rec)
       if (seg = x.AsNumber) then
        jtxt.WriteElt(seg.asString++sal.asString)
        break
        else
        continue
        end
     end
end

MsgBox.Info("Done Writing Input Block J","")
```

- -

'Script: inputd

```
'This script generates the free form text file used to create
'Input Block D
'The following defaults are set in this script:
'  1.  Presently, this program only considers Field 1 flows -- it does not
'      account for second water column flow or pore water flow
'  2.  The flow is steady state
'Tables needed: Same as inputa and Flow Accumulation Values

theProject = av.GetProject
_dir.asFileName.setCWD

dinptxt = LineFile.Make("d.txt".AsFileName,#FILE_PERM_WRITE)
dyntxt = LineFile.Make("dyn.txt".AsFileName,#FILE_PERM_WRITE)

'Get the flow option desired
lst = {"Flow Option One","Flow Option Two","Flow Option Three"}
iqpt = MsgBox.ChoiceAsString(lst,"Choose your flow option (see Help for explanation)",
"Input File D: Flows")
iiqpt = (lst.Find(iqpt))+1
 if (iiqpt = 3) then
```

```
   ttle = MsgBox.Input("Enter the name of the DYNHYD File to be read","Input File D: Flows",
   "*.HYD")
 else
   ttle = "nothing"
 end
 dyntxt.WriteElt(ttle)
 dinptxt.WriteElt(iiqpt.asString++"0")
'Ask if they are going to simulate dry weather conditions

_drywthr = MsgBox.MiniYesNo("Do you want to simulate dry weather conditions?",TRUE)

'identify the two tables that have the segment information in them
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab
flTable = av.GetProject.FindDoc("Flow Accumulation Values")
flVTab = flTable.GetVTab
runTable = av.GetProject.FindDoc("Runoff Accumulation Values")
runVTab = runTable.GetVTab

srt1Field = flTable.GetVTab.FindField("grid-code")
if (flTable.GetWin.IsOpen) then
  flTable.Sort( srt1Field, FALSE )
end


'Create a new table from the Flow Accumulation values which
'holds the segment flow in m^3/sec

nwflVTab=VTab.MakeNew("flow.dbf".asfilename,dBase)
flList=List.Make
flList.Add(Field.Make("grid-code",#FIELD_BYTE,4,0))
flList.Add(Field.Make("cumm_flow",#FIELD_FLOAT,10,2))
flList.Add(Field.Make("int_flow",#FIELD_FLOAT,10,2))
flList.Add(Field.Make("runoff",#FIELD_FLOAT,10,2))
flList.Add(Field.Make("baseflow",#FIELD_FLOAT,10,2))
if (nwflVTab.CanEdit)then
  nwflVTab.SetEditable(true)
 else
   MsgBox.Info("Can't Edit the Table","Exit")
   exit
end

flListclone = flList.DeepClone
nwflVTab.AddFields(flListclone)
nwflFields = nwflVTab.getFields
nwflTable = Table.Make(nwflVTab)
nwflTable.SetName(nwflVTab.GetName)
```

```
gcField = flVTab.FindField("grid-code")
facField = flVTab.FindField("flow accumulation")
nwgcField = nwflVTab.FindField("grid-code")
nwfacField = nwflVTab.FindField("cumm_flow")
nwintflField = nwflVTab.FindField("int_flow")
newrunField = nwflVTab.FindField("runoff")
newbfField = nwflVTab.FindField("baseflow")


' add the cummulative flow in m^3/sec to the new table
nrec = 0
For Each rec in flVTab
 seg = flVTab.ReturnValue(gcField,rec)
 fac = flVTab.ReturnValue(facField,rec)
 flw = fac*(0.000000317098) 'Conversion of mm/yr-cell to m^3/sec
 nwflVtab.AddRecord
 nwflVTab.SetValue(nwgcField,rec,seg)
 nwflVTab.SetValue(nwfacField,rec,flw)
 nrec = nrec + 1
end

' determine the incremental flow and add it to the new
' table
temp = 0
s = 1
For each i in 1..nrec
For each rec in nwflVtab
    seg = nwflVTab.ReturnValue(nwgcField,rec)
    flw = nwflVTab.ReturnValue(nwfacField,rec)
    if (seg = s) then
     int_flw = flw - temp
     s = s+1
     temp = flw
     nwflVTab.SetValue(nwintflField,rec,int_flw)
    end
end
end

'determine the incremental runoff and add that to the new table
rungcField = runVTab.FindField("grid-code")
racField = runVTab.FindField("rfac")

temp = 0
s = 1
for each i in 1..nrec
For each rec in runVTab
  seg = runVTab.ReturnValue(rungcField,rec)
  rac = runVTab.ReturnValue(racField,rec)
  if (seg = s) then
  runoff = (rac-temp)*(0.000000317098)
  s = s+1
  temp = rac
  nwflVTab.SetValue(newrunField,rec,runoff)
  end
end
end

'determine the baseflow into each segment from the total flow
'minus the runoff

For each rec in nwflVTab
  flow = nwflVTab.ReturnValue(nwintflField,rec)
  runoff = nwflVTab.ReturnValue(newrunField,rec)
  baseflow = flow - runoff
  nwflVTab.SetValue(newbfField,rec,baseflow)
end

'join the tables (the new flow table with the main
'segmentation table)
segFields = segVTab.GetFields
jtofield = segFields.Get(5)
jfromfield = nwflFields.Get(0)
segVTab.Join(jtofield,nwflVTab,jfromfield)


typField = bcVTab.FindField("type")
bcdwnField = bcVTab.FindField("dwnstr_seg")
bcgcField = bcVTab.FindField("grid-code")
bcflField = bcVTab.FindField("flow")
bcupField = bcVTab.FindField("upstr_seg")


mngcField = segVTab.FindField("grid-code")
dwnField = segVTab.FindField("dwnstr_seg")
upField = segVTab.FindField("upstr_seg")
incflwField = segVTab.FindField("int_flow")


if (_drywthr = TRUE) then
  wrteField = segVTab.FindField("baseflow")
  else
  wrteField = segVTab.FindField("int_flow")
end

numrecs = segVTab.GetNumRecords
temp = 0
ultdwnstr = 0
For each rec in bcVTab
  seg = bcVTab.ReturnValue(bcgcField,rec)
  dwnstrseg = bcVTab.ReturnValue(bcdwnField,rec)
  if (dwnstrseg = 0) then
  temp = rec
  ultdwnstr = seg
  break
  end
  upstrseg = bcVTab.ReturnValue(bcupField,rec)
  dinptxt.WriteElt(upstrseg.AsString++seg.AsString)
  dinptxt.WriteElt(seg.AsString++dwnstrseg.AsString)
 'Find the corresponding flow for this input
    For each flrec in segVTab
      newseg = segVTab.ReturnValue(mngcField,flrec)
      if (newseg = dwnstrseg) then
        flow = segVTab.ReturnValue(wrteField,flrec)
          else
        continue
      end
    end
 'Find the flow path segment to downstream segment and so on
  For each i in 1..numrecs
      For each n in segVTab
        seg = segVTab.ReturnValue(mngcField,n)
        if (seg = dwnstrseg) then
          dwnstrseg = segVTab.ReturnValue(dwnField,n)
          dinptxt.WriteElt(seg.AsString++dwnstrseg.AsString)
          else
          continue
        end
       end
   end
   dinptxt.WriteElt(dwnstrseg.AsString++"0")
   dinptxt.WriteElt(flow.AsString++"999")
end

'Now, take care of the very last segment
flow = bcVTab.ReturnValue(bcflField,temp)
if (flow <> 0) then
  dinptxt.WriteElt("0"++ultdwnstr.AsString)
  dinptxt.WriteElt(ultdwnstr.AsString++"0")
  dinptxt.WriteElt(flow.AsString++"999")
end

dinptxt.WriteElt("555"++"0")
MsgBox.Info("Done writing Input Block D","")
```

- -

'Script: inpute

```
'This script generates the free form text file needed to complete
'the Input Block E in EUTRO
'The following defaults are set:
 '  1.  Since, presently, only Simple Streeter Phelps is possible,
'      only BOD and DO are considered in the b.c.
'  2.  The b.c.'s are steady state
'Tables needed: same as inputa

theProject = av.GetProject
_dir.asFileName.setCWD

etxt = LineFile.Make("e.txt".AsFileName,#FILE_PERM_WRITE)

if (_imdl = 0)
then
nh3 = 0
n03 = 0
po4 = 0
chla = 0
on = 0
op = 0
end

bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

numrecs = bcVTab.GetNumRecords
typField = bcVTab.FindField("type")
gcField = bcvTab.FindField("grid-code")
doField = bcvTab.FindField("int_do")
bodField = bcvTab.FindField("int_bod")

'Determine the boundary conditions which are water boundary segments

for each rec in bcVTab
  type = bcVTab.ReturnValue(typField, rec)
  if (type = 3) then
   numrecs = numrecs - 1
  end
end

etxt.WriteElt(numrecs.asString++"0")

'get the BOD conc for each boundary

for each rec in bcVTab
  seg = bcVTab.ReturnValue(gcField,rec)
  bod = bcVTab.ReturnValue(bodField,rec)
  type = bcVtab.ReturnValue(typField,rec)
  if (type = 3)
  then
  break
  end
  etxt.WriteElt(seg.asString++bod.asString)
end

'get the DO for each boundary

for each rec in bcVTab
  seg = bcVTab.ReturnValue(gcField,rec)
  do = bcVTab.ReturnValue(doField,rec)
  type = bcVtab.ReturnValue(typField,rec)
  if (type = 3)
  then
  break
  end
  etxt.WriteElt(seg.asString++do.asString)
end

MsgBox.Info("Done writing Input Block E","")
```

. . . .

'Script: inputf

```
'This script creats the free form text file needed to generate
'Input Block F
'The following defaults are set:
'  1.  All point and non-point sources are steady state
'  2.  Since presently, the loads are steady state, the nps
'      loads are added to the point source loads and written
'      in the main input block; if unsteady nps loads are
'      determined, a new file must be created, separate from
'      the main input file in WASP -- note that this fact would
'      entail some changes in the FORTRAN code which formats the
'      input file (ougen.for)

'Tables Needed: same as inputa and Runoff Accumulation Values,
'               Point Source BOD, and BOD Loading Values

if (_drywthr = nil) then
_drywthr = MsgBox.MiniYesNo("Do you want to simulate dry weather conditions?",TRUE)
end

theProject = av.GetProject
_dir.asFileName.setCWD

ftxt = LineFile.Make("f.txt".AsFileName,#FILE_PERM_WRITE)

'identify the two tables that have the segment information in them
psTable = av.GetProject.FindDoc("Point Source BOD")
psVTab = psTable.GetVTab
npsTable = av.GetProject.FindDoc("BOD Loading Values")
npsVTab = npsTable.GetVTab
runTable = av.GetProject.FindDoc("Runoff Accumulation Values")
runVTab = runTable.GetVTab
```

```
'Create a Table which will hold the Loadings in kg/day
ldVTab=VTab.MakeNew("load.dbf".asfilename,dBase)
ldList=List.Make
ldList.Add(Field.Make("grid-code",#FIELD_BYTE,4,0))
ldList.Add(Field.Make("bod_ps",#FIELD_FLOAT,10,2))
ldList.Add(Field.Make("bod_nps",#FIELD_FLOAT,10,2))

if (ldVTab.CanEdit)then
  ldVTab.SetEditable(true)
 else
   MsgBox.Info("Can't Edit the Table","Exit")
   exit
end

ldListclone = ldList.DeepClone
ldVTab.AddFields(ldListclone)
ldFields = ldVTab.getFields
ldTable = Table.Make(ldVTab)
ldTable.SetName(ldVTab.GetName)
ldgcField = ldVTab.FindField("grid-code")
psField = ldVTab.FindField("bod_ps")
bodnpsField = ldVTab.FindField("bod_nps")

'Find the number of main segments that get loading and
'write it to the file

numrecs = npsVTab.GetNumRecords
ftxt.WriteElt(numrecs.asString++"0")

'Find the loadings from each table:
'  1. BOD Point Sources
'  2. BOD NPS from BOD Flow Accumulation
'Write these loads to the table "load.dbf" and
'to the file f.txt

gcField = psVTab.FindField("segment")
lbpsField = psVTab.FindField("bod")
runField = runVTab.FindField("rfac")
bodField = npsVTab.FindField("accumulated bod loading")

for each rec in psVTab
  seg = psVTab.ReturnValue(gcField,rec)
  load = psVTab.ReturnValue(lbpsField,rec)
  load = load * 1.243
  ftxt.WriteElt(seg.asString++load.asString)
  ldVTab.AddRecord
  ldVTab.SetValue(ldgcField,rec,seg)
  ldVTab.SetValue(psField,rec,load)
end

bodlst = List.Make
for each rec in npsVTab
  load = npsVTab.ReturnValue(bodField,rec)
  bodlst.Add(load)
end
bodlst.Sort(TRUE)

p = numrecs+1
n = 0
s = 1
temp = 0
pre = 0
for each i in bodlst
  temp = bodlst.Get(n)
  load = temp - pre
  load = load/365
  if (_drywthr = TRUE) then
    ftxt.WriteElt(s.AsString++"0")
  else
  ftxt.WriteElt(s.AsString++load.AsString)
  end
    for each i in 1..p
      for each rec in ldVTab
        x = ldVTab.ReturnValue(ldgcField,rec)
        if (x=s) then
          ldVTab.SetValue(bodnpsField,rec,load)
          break
        end
       end
     end
    pre = temp
    n = n+1
    s = s+1
end


MsgBox.Info("Done writing Input Block F","")
```

- -

```
'Script: inputg
'This script creates a the free form text file used to generate
'the input block G
'The following defaults are set in this file:
'  1.  Theta does not spatially vary
'Tables needed:  Same as inputa

theProject = av.GetProject
_dir.asFileName.setCWD
```

```
gtxt = LineFile.Make("g.txt".AsFileName,#FILE_PERM_WRITE)

'get the important tables
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

gcField = segVTab.FindField("grid-code")
salField = segVTab.FindField("sal")
tempField = segVTab.FindField("temp")
sodField = segVTab.FindField("sod")

bcgcField = bcVTab.FindField("grid-code")
bcsalField = bcVTab.FindField("sal")
bctempField = bcVTab.FindField("temp")
bcsodField = bcVTab.FindField("sod")

'Get the theta used for the SOD temperature correction (spatially constant)

sodta = MsgBox.Input("Please enter the Theta used to SOD temperature correction","Input File G:
Parameters","1.065")
gtxt.WriteElt(sodta.asString++"0"++"0"++"0")

'Get the temperature, salinity, and sediment oxygen demand for each segment
'and write it to the input file

for each rec in segVTab
  seg = segVTab.ReturnValue(gcField,rec)
  tmp = segVTab.ReturnValue(tempField,rec)
  sal = segVTab.ReturnValue(salField,rec)
  sod = segVTab.ReturnValue(sodField,rec)
  gtxt.WriteElt(seg.asString++tmp.asString++sod.asString++sal.asString)
end

for each rec in bcVTab
  seg = bcVTab.ReturnValue(bcgcField,rec)
  tmp = bcVTab.ReturnValue(bctempField,rec)
  sal = bcVTab.ReturnValue(bcsalField,rec)
  sod = bcVTab.ReturnValue(bcsodField,rec)
  gtxt.WriteElt(seg.asString++tmp.asString++sod.asString++sal.asString)
end

MsgBox.Info("Done writing Input Block G","")
```

- -

'Script: inputh

```
'This file obtains the constants necessary to run the WASP program
'for the Input Block H
'Defaults:
'Since only Simple Streeter Phelps is possible, the only two
'constants needed are the deoxygenation rate and the
'reaeration rate
'NOTE:  This connection assumes that the reaeration rate is constant
'over time and space
'Tables Needed: NONE

theProject = av.GetProject
_dir.asFileName.setCWD

htxt = LineFile.Make("h.txt".AsFileName,#FILE_PERM_WRITE)

'Get the constants Kd and Ka from the user and write them to a file

lst = {"CBOD Deoxygenation Coefficient (/day @ 20°C)","Reaeration Rate (/day)"}
lstdef = {"0.1","0.1"}
cnst = MsgBox.MultiInput("Please Enter the Following Constants","Input File H: Constants",lst,lstdef)

for each i in cnst
  htxt.WriteElt(i.AsString)
end

MsgBox.Info("Done writing Input Block H","")
```

- -

'Script: inputi

```
'This script writes the free from text file used to
'create the Input Block I
'NOTE:  Since this connection is presently set up for
'just steady state, no Time Functions are necessary
'Tables needed: NONE

theProject = av.GetProject
_dir.asFileName.setCWD

itxt = LineFile.Make("i.txt".AsFileName,#FILE_PERM_WRITE)

MsgBox.Info("There are no time functions needed for this current model","Input File I: Time Functions")
itxt.WriteElt("0")
```

- -

'Script: inputj

```
'This script writes the text file used to create
'the Input Block J for WASP
'This script presently only reads initial conditions for bod and do
'Tables needed: same as inputa

theProject = av.GetProject
_dir.asFileName.setCWD
```

```
jtxt = LineFile.Make("j.txt".AsFileName,#FILE_PERM_WRITE)

'get the necessary tables
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

gcField = segVTab.FindField("grid-code")
doField = segVTab.FindField("int_do")
bodField = segVTab.FindField("int_bod")
bcgcField = bcVTab.FindField("grid-code")
bcdoField = bcVTab.FindField("int_do")
bcbodField = bcVTab.FindField("int_bod")

'Write 0 for initial condition for systems 1 through 4

for each i in 1..4 'nh3,n03,po4,chl
jtxt.WriteElt("1"++"0")
n=1
totrec = segVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in segVTab
      seg = segVTab.ReturnValue(gcField,rec)
        if (seg = n) then
         jtxt.WriteElt(seg.asString++"0")
         n= n+1
         break
         end
     end
end

totrec = bcVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in bcVTab
      seg = bcVTab.ReturnValue(bcgcField,rec)
        if (seg = n) then
         jtxt.WriteElt(seg.asString++"0")
         n= n+1
         break
         end
     end
end

end

'obtain the BOD initial conditions and write them, in
'ascending order into the text file
jtxt.WriteElt("0.5"++"0") 'writing the dissolved fraction
n=1
totrec = segVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in segVTab
      seg = segVTab.ReturnValue(gcField,rec)
      bod = segVTab.ReturnValue(bodField,rec)
        if (seg = n) then
         jtxt.WriteElt(seg.asString++bod.asString)
         n= n+1
         break
         end
     end
end

totrec = bcVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in bcVTab
      seg = bcVTab.ReturnValue(bcgcField,rec)
      bod = bcVTab.ReturnValue(bcbodField,rec)
        if (seg = n) then
         jtxt.WriteElt(seg.asString++bod.asString)
         n= n+1
         break
         end
     end
end

'obtain the DO initial conditions and write them, in
'ascending order into the text file
jtxt.WriteElt("1"++"0")
n=1
totrec = segVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in segVTab
      seg = segVTab.ReturnValue(gcField,rec)
      do = segVTab.ReturnValue(doField,rec)
        if (seg = n) then
         jtxt.WriteElt(seg.asString++do.asString)
         n= n+1
         break
         end
     end
end

totrec = bcVTab.GetNumRecords
```

```
p = totrec + 1
  for each i in 1..p
    for each rec in bcVTab
      seg = bcVTab.ReturnValue(bcgcField,rec)
      do = bcVTab.ReturnValue(bcdoField,rec)
       if (seg = n) then
        jtxt.WriteElt(seg.asString++do.asString)
         n= n+1
        break
        end
     end
end

'writing 0 for initial conditons of system 7 and 8
for each i in 1..4 'on,op
jtxt.WriteElt("1"++"0")
n=1
totrec = segVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in segVTab
      seg = segVTab.ReturnValue(gcField,rec)
       if (seg = n) then
        jtxt.WriteElt(seg.asString++"0")
         n= n+1
        break
        end
     end
end

totrec = bcVTab.GetNumRecords
p = totrec + 1
  for each i in 1..p
    for each rec in bcVTab
      seg = bcVTab.ReturnValue(bcgcField,rec)
       if (seg = n) then
        jtxt.WriteElt(seg.asString++"0")
         n= n+1
        break
        end
     end
end

end

MsgBox.Info("Done writing Input Block J","")
```

. . . .

'Script: mod_parchk

```
'this script checks the model parameters for the eutro
'output file, beofre the user can run the output
'processer

theProject = av.GetProject

starttxt = LineFile.Make("estart.txt".AsFileName,#FILE_PERM_WRITE)
_dir = MsgBox.Input("Enter the working directory.","Working Directory","c:\benaman\wasp")
_dir.AsFilename.setCWD


segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab
bcTable = av.GetProject.FindDoc("Boundary Segments")
bcVTab = bcTable.GetVTab

_seg = (segVTab.GetNumRecords)+(bcVTab.GetNumRecords)

_dts = MsgBox.Input("What was the maximum time step allowed (/day)?","Input File A: Model Options","0.001")

_tend = MsgBox.Input("How many days did you run the model?","Input File A: Model Options","100.0")

_prn = MsgBox.Input("At what interval (in days) did you have the results printed to the output file?","Input File A:
Model Options","1.0")

outname = MsgBox.Input("Enter the name of the file you want to process","Model Output Processing","*.edf")
outtxt = LineFile.Make("eoutnme.txt".AsFileName,#FILE_PERM_WRITE)
outtxt.WriteElt(outname)

qu = MsgBox.YesNo("Is the output file called"++outname+"?","Calibration Output Processing",TRUE)
if (qu=TRUE) then
lst = "Is the following correct?"+nl+"Number of Days Run:"++_tend.AsString++"days"+nl+"Print Interval for output
file:"++_prn.AsString++"days"+nl+"Number of Segments:"++_seg.AsString
qu2 = MsgBox.YesNo(lst,"Model Output Processing",TRUE)
  if (qu2=TRUE) then
  starttxt.WriteElt(_tend.AsString++_dts.AsString++_prn.AsString++_seg.AsString)
  else
  MsgBox.Info("Please see the Help for how to set up the output for processing","")
  end
else
MsgBox.Info("Please see the Help for how to set up the output for processing","")
exit
end
```

. . . .

'Script: run_calout

```
'This Script will process the TOXI5 tdf file into a
'text file, containing an array of time vs. salinity
'by segment number

'executing the fortran program which extracts the salinity measurements
```

```
'from the tdf file
_dir.asFileName.setCWD

MsgBox.Info("Press 'OK' to begin","")
system.execute("calout.exe")
MsgBox.Info("Click 'OK' when done processing calibration output","")

'making a table, named by the user, which will hold the output
name = MsgBox.Input("Name the Table (no more than 8 characters)","Calibration View Output","salinity")

theProject=av.GetProject

outfile = linefile.make(("salinity.txt").asFileName,#FILE_PERM_READ)

salVTab=VTab.MakeNew((name+".dbf").asFileName,dBase)
salList=List.Make
salList.Add(Field.Make("Time",#FIELD_FLOAT,15,4))
for each i in 1.._seg
salList.Add(Field.Make((i.asString),#FIELD_FLOAT,10,4))
end

if (salVTab.CanEdit) then
  salVTab.SetEditable(true)
else
  MsgBox.Info("Can't Edit","Exit")
  exit
end

salclneList = salList.DeepClone
salVTab.AddFields(salclneList)
salFields=salVTab.getFields
salTable=Table.Make(salVTab)
salTable.SetName(salVTab.GetName)

'reading the text file, calout.txt, and
'importing the values into salinity.dbf

while (outfile.IsAtEnd=FALSE)
  newrec=salVTab.AddRecord
  inline = outfile.ReadElt
  count = 0
  for each f in salFields
    val=inline.Extract(count).AsNumber
    salVTab.SetValue(f,newrec,val)
    count = count + 1
   end
end

MsgBox.Info("Done making salinity table","")
```

- -

'Script: run_modout

```
'This Script will process the EUTRO5 edf file into a
'text file, containing an array of time vs. salinity
'by segment number

'executing the fortran program which extracts the salinity measurements
'from the tdf file
_dir.asFileName.setCWD

MsgBox.Info("Press 'OK' to begin","")
system.execute("modout.exe")
MsgBox.Info("Click 'OK' when done processing calibration output","")

'making a table, named by the user, which will hold the output
bodname = MsgBox.Input("Name the Table for BOD (no more than 8 characters)","View Output","bod")
doname = MsgBox.Input("Name the Table for DO (no more than 8 characters)","View Output","do")

theProject=av.GetProject

outfile1 = linefile.make(("bod.txt").asFileName,#FILE_PERM_READ)
outfile2 = linefile.make(("do.txt").asFileName,#FILE_PERM_READ)

'Making bod table
bodVTab=VTab.MakeNew((bodname+".dbf").asFileName,dBase)
salList=List.Make
salList.Add(Field.Make("Time",#FIELD_FLOAT,15,4))
for each i in 1.._seg
salList.Add(Field.Make((i.asString),#FIELD_FLOAT,10,4))
end

if (bodVTab.CanEdit) then
  bodVTab.SetEditable(true)
else
  MsgBox.Info("Can't Edit","Exit")
  exit
end

bodclneList = salList.DeepClone
bodVTab.AddFields(bodclneList)
bodFields=bodVTab.getFields
bodTable=Table.Make(bodVTab)
bodTable.SetName(bodVTab.GetName)

'reading the text file, bod.txt, and
'importing the values into bod.dbf

while (outfile1.IsAtEnd=FALSE)
  newrec=bodVTab.AddRecord
  inline = outfile1.ReadElt
  count = 0
  for each f in bodFields
    val=inline.Extract(count).AsNumber
    bodVTab.SetValue(f,newrec,val)
    count = count + 1
   end
end

MsgBox.Info("Done making BOD table","")

'Making do table
doVTab=VTab.MakeNew((doname+".dbf").asFileName,dBase)
doList=List.Make
doList.Add(Field.Make("Time",#FIELD_FLOAT,15,4))
for each i in 1.._seg
doList.Add(Field.Make((i.asString),#FIELD_FLOAT,10,4))
end

if (doVTab.CanEdit) then
  doVTab.SetEditable(true)
else
  MsgBox.Info("Can't Edit","Exit")
  exit
end

doclneList = doList.DeepClone
doVTab.AddFields(doclneList)
doFields=doVTab.getFields
doTable=Table.Make(doVTab)
doTable.SetName(doVTab.GetName)

'reading the text file, do.txt, and
'importing the values into do.dbf

while (outfile2.IsAtEnd=FALSE)
  newrec=doVTab.AddRecord
  inline = outfile2.ReadElt
  count = 0
  for each f in doFields
    val=inline.Extract(count).AsNumber
    doVTab.SetValue(f,newrec,val)
    count = count + 1
   end
end

MsgBox.Info("Done making Dissolved Oxygen table","")
```

. . . .

'Script: toxirun

```
'This script executes the WASP model TOXI5
'it allows you to choose any input file, but the one
'generated by this connection will be called "caltest.inp"

_dir.asFileName.setCWD
system.execute("toxi5.exe")
MsgBox.Info("Click 'OK' when TOXI5 is done running","")
```

. . . .

'Script: vwout

```
'This script controls the output options

theProject = av.GetProject

' Creates a list of tables and allows the user to pick which one they
' want to view the output for.
docList = theProject.GetDocs
tabList = List.Make
numdocs=docList.count
for each i in 0..(numdocs-1)
  dtype=(docList.get(i)).GetClass.GetClassName
  if (dtype="Table") then
    tabList.Add(docList.Get(i).GetName)
  end
end
_tblname = MsgBox.ChoiceAsString(tabList,"Choose the output table you want to work with","View Output")


'Sets the tables and important themes needed for this script
salTable = av.GetProject.FindDoc(_tblname)
salVTab = salTable.GetVTab
salFields = salVTab.GetFields
timeField = salVtab.FindField("time")
theView = av.GetProject.FindDoc("Segmentation")
segTheme = theView.FindTheme("Main Segmentation")
bcTheme = theView.FindTheme("Boundary Segmentation")
segTable = av.GetProject.FindDoc("Main Segmentation")
segVTab = segTable.GetVTab


'Checks to make sure the table selected has a time index to plot
if (timeField=nil) then
  MsgBox.Info("This table does not include a time index","")
  exit
end

mnsegs = segVTab.GetNumRecords

'Prompts the user to choose which type of output they want to view
for each rec in salVTab
  tnd = SalVTab.ReturnValue(timeField,rec)
```

```
end

lst = {"Table: Conc vs. t","Chart: Conc vs. t for one Segment","Chart: Conc vs. Seg# for a given Time","Coverage: Conc
at t = "++tnd.AsString++" days","Movie:  Concentration over time"}
vw = MsgBox.ChoiceAsString(lst,"Choose the output you wish to view","View Output")
ivw = lst.Find(vw)

myBitMap = salVTab.GetSelection
myBitMap.ClearAll

'open the table selected
if (ivw = 0) then
  salTable.GetWin.Open

'creates a chart of conc vs. time for a segment chosen from the view
elseif(ivw = 1) then
  myBitMap.ClearAll
  ms = "Have the bug icon active,"
  ms = ms+nl+"Have the segmentation theme active,"
  ms = ms+nl+"Select the segment to graph"
  MsgBox.Info(ms,"View Output")
  theView.GetWin.Open
  theView.GetWin.Maximize

'creates a chart of concentration vs. Segment Number for a given time
elseif (ivw = 2) then

'Allows the user to choose the color of the chartcolorlist=List.Make
colorlist = List.Make
colorlist.Add("blue")
colorlist.Add("yellow")
colorlist.Add("green")
colorlist.Add("red")
col=MsgBox.ChoiceAsString(colorlist,"Select a color for the chart","View Output")
if (col="blue") then
  _chcolor = Color.GetBlue
elseif (col="yellow") then
  _chcolor = Color.GetYellow
elseif (col="green") then
  _chcolor = Color.GetGreen
else (col="red")
  _chcolor = Color.GetRed
end

   timelst = List.Make
   For each rec in salVTab
     time = salVTab.ReturnValue(timeField,rec)
     timelst.Add(time.AsString)
   end
   t = MsgBox.ChoiceAsString(timelst,"Choose the time (in days) you would like to graph.","View Output")
   timerec = timelst.Find(t)
   sallst = List.Make
   For each i in 1.._seg
     sallst.Add(salFields.Get(i))
   end

   myBitMap.Set(timerec)
   salVTab.UpdateSelection
   xChart = Chart.Make(salVTab,sallst)
   xchartname=xchart.getname
   theProject.setActive(xchart)
   xChartDisp = xchart.GetChartDisplay
   xChartDisp.setType(#CHARTDISPLAY_column)
   xChartDisp.SetSeriesColor(0,_chcolor)
   the_x=xChart.GetXAxis
   the_y=xChart.GetYAxis
   the_x.SetTickLabelsVisible(True)
   the_x.SetMajorGridVisible(True)
   the_y.SetMajorGridVisible(True)
   the_x.SetCrossValue(0)
   the_y.SetCrossValue(0)
   the_x.SetLabelVisible(true)
   the_y.SetLabelVisible(true)
   xLegend=xChart.GetChartLegend
   xLegend.SetVisible(False)
   xChart.GetTitle.SetName("Day = "++t.AsString)
   ylst = {"Salinity (ppt)","DO (mg/L)","BOD (mg/L)"}
   yname = MsgBox.ChoiceAsString(ylst,"Please enter the y-axis","View Output")
   the_x.SetName("Segment Number")
   the_y.SetName(yname)
   xchart.GetWin.Open
   keep = MsgBox.YesNo("Would you like to keep the chart?","ArcView",true)
   if (keep=false) then
     av.GetProject.removeDoc(xchart)
     exit
   else
     chname = MsgBox.Input("Please name the chart window","View Output","")
     xchart.setname(chname)
   end

'creates a coverage of concentration and puts it on the active view
elseif (ivw=3) then
  'Creates a new theme on the View
  myBitMap.ClearAll
  theView.GetWin.Open
  theView.GetWin.Maximize
  segSrcName = SrcName.Make("c:\wasp\cover\segarc arc")
  newTheme = Theme.Make(segSrcName)
  newTheme.SetVisible(true)
  theView.AddTheme(newTheme)
  theView.Invalidate
  newFtab = newTheme.GetFTab
  'sets the name of the Theme
  thmename = MsgBox.Input("Name the new coverage","View Output","")
  newTheme.SetName(thmename++"@ Time = "+tnd.AsString)

'counts the number of records in the salVTab
    cntr = 0
  for each rec in salVTab
   cntr = cntr + 1
  end

  'create a table that will be joined to the aat
  'of the new theme and holds the concentration values
  cncVTab = VTab.MakeNew("temp.dbf".asFileName,dBase)
  cncLst=List.Make
  cncLst.Add(Field.Make("grid-code",#FIELD_BYTE,4,0))
  cncLst.Add(Field.Make("conc",#FIELD_FLOAT,10,2))
  cncVTab.SetEditable(true)

  cncLstClone = cncLst.DeepClone
  cncVTab.AddFields(cncLstClone)
  cncFields = cncVTab.getFields

  cncTable = Table.Make(cncVTab)
  cncTable.SetName(cncVTab.GetName)
  cncgcField = cncVTab.FindField("grid-code")
  cncField = cncVTab.FindField("conc")
  cntr = cntr-1
  for each i in 1..mnsegs
    rec = i-1
    cncVTab.AddRecord
    cncVTab.SetValue(cncgcField,rec,i.AsString)
    newField = salFields.Get(i)
    val = salVTab.ReturnValue(newField,cntr)
    cncVTab.SetValue(cncField,rec,val)
  end

  newFields = newFTab.GetFields
  jtofield = newFields.Get(8)
  jfromField=cncFields.Get(0)
  newFTab.Join(jtofield,cncVTab,jfromfield)
  lookField = newFTab.FindField("conc")
  theLegend = newTheme.GetLegend
  theLegend.Quantile(newFTab,lookField,6)
  theLegend.RampColors(Color.GetBlue,Color.GetRed)
  for each i in theLegend.GetSymbols
    i.SetWidth(3)
  end
  newTheme.UpdateLegend

  av.GetProject.removeDoc(cncTable)

'Creates a 'movie' of four frames at chosen times
  elseif (ivw=4) then
  theView = theProject.FindDoc("Segmentation")
  m = "This option allows you to choose four times and view how the concentrations change, over time."
  MsgBox.Info(m,"View Output")
  timelst = List.Make
   For each rec in salVTab
     if (rec = 0) then
     continue
     else
     time = salVTab.ReturnValue(timeField,rec)
     timelst.Add(time.AsString)
     end
   end
   for each i in 1..4
   t = MsgBox.ChoiceAsString(timelst,"Choose the time"++i.asString++"(days)","View Output")
   if (i = 1) then
   t1 = timelst.Find(t)+1
   time1 = salVTab.ReturnValue(timeField,t1)
   elseif (i=2) then
   t2 = timelst.Find(t)+1
   time2 = salVTab.ReturnValue(timeField,t2)
   elseif (i=3) then
   t3=timelst.Find(t)+1
   time3 = salVTab.ReturnValue(timeField,t3)
   elseif (i=4) then
   t4 = timelst.Find(t)+1
   time4 = salVTab.ReturnValue(timeField,t4)
   end
   end


   segSrcName = srcName.Make("c:\wasp\cover\segarc arc")
   concname = MsgBox.Input("Name the concentration you are viewing","View Output","")
   s = MsgBox.Input("Choose a step time (in sec) for the movie frames","View Output","5")
   s = s.AsNumber

'creates a table which will hold the conc values for
' all four times

   movVTab = VTab.MakeNew("temp.dbf".asFileName,dBase)
   movLst = List.Make
   movLst.Add(Field.Make("grid-code",#FIELD_BYTE,4,0))
   movLst.Add(Field.Make("conc1",#FIELD_FLOAT,10,2))
   movLst.Add(Field.Make("conc2",#FIELD_FLOAT,10,2))
   movLst.Add(Field.Make("conc3",#FIELD_FLOAT,10,2))
   movLst.Add(Field.Make("conc4",#FIELD_FLOAT,10,2))
   movVTab.SetEditable(true)
```

```
 movLstClone = movLst.DeepClone
 movVTab.AddFields(movLstClone)
 movFields = movVTab.getFields

 movTable = Table.Make(movVTab)
 movTable.SetName(movVTab.GetName)
 movgcField = movVtab.FindField("grid-code")
 c1Field = movVTab.FindField("conc1")
 c2Field = movVtab.FindField("conc2")
 c3Field = movVtab.FindField("conc3")
 c4Field = movVtab.FindField("conc4")
 minLst = List.Make
 for each x in 1..mnsegs
   rec = x-1
   movVTab.AddRecord
   movVTab.SetValue(movgcField,rec,x.AsString)
   newField = salFields.Get(x)
   val1 = salVTab.ReturnValue(newField,t1)
   val2 = salVTab.ReturnValue(newField,t2)
   val3 = salVTab.ReturnValue(newField,t3)
   val4 = salVTab.ReturnValue(newField,t4)
   movVTab.SetValue(c1Field,rec,val1)
   movVTab.SetValue(c2Field,rec,val2)
   movVTab.SetValue(c3Field,rec,val3)
   movVTab.SetValue(c4Field,rec,val4)
   minLst.Add(val1)
   minLst.Add(val2)
   minLst.Add(val3)
   minLst.Add(val4)
 end

n = 35
x = 0
for Each m in MinLst
  if (m < n) then
    n = m
  end
end

for Each m in MinLst
  if (m > x) then
    x = m
  end
end

inv = (x - n)/8

   theView.GetWin.Open
   theView.GetWin.Maximize
   av.delayedrun("frame1",{segSrcName,concname,time1,theView,movVtab,x,n,inv},25)
   av.delayedrun("frame2",{segSrcName,concname,time2,theView,movVtab,x,n,inv},27+s)
   av.delayedrun("frame3",{segSrcName,concname,time3,theView,movVtab,x,n,inv},s+s+29)
   av.delayedrun("frame4",{segSrcName,concname,time4,theView,movVtab,x,n,inv},s+s+s+31)
   av.GetProject.removeDoc(movTable)
   end
```

Appendix G
Help File for ArcView/WASP5 Connection

created using winhelp.exe standards

WASP5/ArcView Connection

Information and Help

Table of Contents

This ArcView/WASP5 connection was created by Jennifer Benaman, Research Assistant, Department of Civil Engineering, The University of Texas at Austin. Last revised: August 1996.

General Information and Concept

This connection is an ArcView Project which has compiled Avenue Scripts to perform the following functions:

1. Obtains the necessary information needed to run WASP5 and writes it to free form text files. This project will write a text file for each input block. As a result, ten separate text files will be written in the working directory. In addition, three character text files, containing filenames, are created.
2. Runs a FORTRAN program which takes these text files and formats them into the input file for WASP5. The user will be prompted to name the input file.
3. Executes WASP5 (EUTRO5 or TOXI5).
4. Processes the model output.
5. Allows the user to view the output in the form of charts and tables.

Presently, this connection is set up to run a model with the following characteristics:

- Level 1 complexity, EUTRO5 model (Simple Streeter-Phelps Model for BOD/DO)
- Steady-state conditions (Flow conditions: Average Year or Dry Weather)
- System should resemble river or stream (i.e. main model segments have dominating flow in one direction and have exchange with other main segments at the upstream or downstream ends).
- Only water column flow is considered presently. This connection is not set up to deal with pore water column flow, secondary water column flow, or sediment transport. The user can have a single benthic layer under the water segments as a boundary condition to deal with settling.

In addition, this connection can run TOXI5 to calibrate the system, by using salinity as the conservative substance. It is recommended that the model be calibrated, before an actual BOD/DO model run is performed.

If the user has additional questions concerning the model parameters or variables used in the input file, please refer to the WASP5 User's Manuals (A and B).

Related Topics:

About WASP5

About this Connection

Limitations and Important Notes Concerning this Connection

Table of Contents

Setting up the WASP5/ArcView Connection

The ArcView/WASP5 model connection and a demo which shows the Houston Ship Channel study discussed in this report can be set up on any computer which has ArcView 2.1 or higher installed on the machine. In order to set up the demonstration on a personal computer, the following steps are taken:

1. Install WASP5 onto the computer. WASP5 is available from the USEPA Homepage (ftp://ftp.epa.gov/epa_ceam/wwwhtml/wasp.htm)

2. Download the demonstration files, in zipped format, from the University of Texas, Center for Research in Water Resources Homepage for the ArcView/WASP5 connection demo (http://www.ce.utexas.edu/prof/maidment /GISHydro/) and unzip them into the directory that the WASP5 executables are located.

3. Open the project hsc_wasp.apr in ArcView version 2.1 or higher. ArcView may initially ask the user to location of some coverages and tables. All necessary coverages and tables should be located with in the directory in which the downloaded file was unzipped. Most tables are in dbf format, while the coverages are in a folder entitled *cover*. Open the script *vwout* and locate the two references to the arc coverage *segarc*. Both of the lines in the script read:

segSrcName = SrcName.Make("c:\wasp\cover\segarc arc")

This line informs Avenue on where to locate the main segmentation coverage. Be sure that the drive and directory name in this line is correct. Also, if this script is being recompiled for a new modeling system, this line should reference the correct location and name for the main segmentation coverage in the new system that has been developed.

4. If the *vwout* script was changed, in any way, recompile it, by clicking on the checkmark icon on the bottom toolbar of the ArcView script tools.

5. Model input file creation, model runs, and output viewing can then be performed as described previously in this help file.

Related Topics:

Coverages Needed for Processing

Table of Contents

Tables Needed for Processing

It is extremely important that all tables listed here are included and opened in the project, before the model input file can be created. It is also very important that all tables and table fields are named (either by alias or real name) EXACTLY as they are written here.

NOTES:

- Tables are listed here in alphabetical order.

-The order of the records is not important.

-The order of the fields should be adhered to in the Main Segmentation Table (grid-code should always be the 6th field) and in the Main Segment Parameters Table (grid-code should always be the first field).

- Weighted flow accumulation values are based on flow accumulations run on 100m x 100m cell grids. The unit conversion set in the scripts are also based on this cell-size.

Click on the table name to get a list of fields and their corresponding units.

BOD Loading Values

A table (typically a value attribute table -- INFO format) of the BOD flow accumulation values for each "outlet" (i.e. its most downstream point) of the main water segments.   These values are usually obtained by running a flow accumulation over the watershed area, weighted by a grid of BOD load, in ArcInfo's subprogram, Grid. This weighted flow accumulation is then "combined" with a grid of the outlet points to obtain this table. This table accounts for the non-point source loading from the watershed land surface.

Boundary Segments

A dBase file (dbf) table which contains all boundary segments (water and sediment), and their corresponding parameters. This file can be created in directly in dBase, or in ArcInfo and exported out of ArcView to a dBase format. If the table is not a dBase format, it will not be possible to edit and change parameters in the table. *The numbering of the boundary segments should start which the next number after the last main segment.*

Flow Accumulation Values

A table (typically a value attribute table -- INFO format) of the flow accumulation values for each "outlet" (i.e. its most downstream point) of the main water segments. These values are usually obtained by running a flow accumulation over the watershed area, weighted by a grid of flow depth, in ArcInfo's subprogram, Grid. This weighted flow accumulation is then "combined" with a grid of the outlet points to obtain this table.

Main Segment Parameters

A dbf file which contains the attributes of the main segments for the water quality model. This file can be created in directly in dBase, or in ArcInfo and exported out of ArcView to a dBase format. If the table is not a dBase format, it will not be possible to edit and change parameters in the table. This table will be joined, during the generation of Input Block A, to the "Main Segmentation" table below. *The numbering of the main segments should start with "1" and continue, in order, until the last main segment is numbered. Then, number the boundary segments.*

Main Segmentation

The aat of the main segmentation coverage. *The numbering of the main segments should start with "1" and continue, in order, until the last main segment is numbered. Then, number the boundary segments.*

Point Source BOD

A dbf file which contains the annual BOD loading into each segment from point sources. This file can be created in directly in dBase, or in ArcInfo and exported out of ArcView to a dBase format. If the table is not a dBase format, it will not be possible to edit and change parameters in the table.

Runoff Accumulation Values

A table (typically a value attribute table -- INFO format) of the flow accumulation values for each "outlet" (i.e. its most downstream point) of the main water segments. These values are usually obtained by running a flow accumulation over the watershed area, weighted by a grid of runoff depth, in ArcInfo's subprogram, Grid. This weighted flow accumulation is then "combined" with a grid of the outlet points to obtain this table.

Water Boundary Segmentation

The arc attribute table (aat) of the water boundary segment reaches. It is important NOT to join this table with the "Boundary Segments" table, since the boundary segments table may have segments which are not Type 1 (i.e. Water Column). If joined, segments shown on the Boundary Segments table which are not represented in the coverage as water reaches will be "lost". *The numbering of the boundary segments should start with the next number after the last main segment.*

Related Topics:

Coverages Needed for Processing

Table of Contents

Coverages Needed for Processing

Although the user may wish to add other coverages for informational purposes, there are only two arc coverages needed to run this model connection. Both should be located within the same view.
The view should be entitled "Segmentation". The two themes are described below:

Boundary Segmentation

An arc coverage of the water boundary segments. The arc attribute table attached to this theme should be named "Water Boundary Segmentation".

Main Segmentation

An arc coverage of the main segmentation being modelled by WASP. The arc attribute table to this theme should be named "Main Segmentation". This coverage should be named "segarc" within ArcInfo and aliased as "Main Segmentation" on the view.

Related Topics:

Tables Need for Processing

Table of Contents

Creating an Input File

The way this connection works is by reading necessary table information and querying the user for needed model options, during the input file generation. WASP5 has ten input blocks (A through J) and a "free form" text file is created for each block. In addition, three text files containing the model run description, DYNHYD file name (if necessary), and the input filename are created. Once the input file generation is performed, these text files will be in the working directory as a.txt, b.txt, etc. It is then possible to change just one input block (i.e. just the model constants -- Input Block H), while the rest of the parameters will stay as they were originally generated.

To create your first input file for your system, follow the following steps:

1. Make sure all FORTRAN programs (calout.exe, calgen.exe, modout.exe, outgen.exe), WASP5 executables, and their related files are located in your working directory.
2. Check the script, vwout, to ensure that the correct coverage and directory is being referenced (see Setting up the ArcView/WASP5 Connection for more information on this step).
3. Have all of the necessary Tables and Views open. They can be minimized to icon views, but they must be open.
4. Have the "Project" window active so that the model connection menu items are shown on the Main Menu Bar. They are: BOD/DO Input Blocks, BOD/DO Model, and Model Calibration.
5. Choose BOD/DO Input Blocks: All Input Blocks and allow the project to run. The entire process will take about 2-3 minutes. The scripts will ask you for some information for some blocks, while other blocks will not require any user input. The project will give you a message box each time it is done writing a particular Input Block. Your working directory should ALWAYS be the directory which holds eutro5.exe, toxi5.exe, outgen.exe, calgen.exe, calout.exe, and modout.exe.
6. Once all ten blocks are written, choose BOD/DO Model: Generate Input File.
7. Be sure you have calibrated your model before running the BOD/DO model.

IMPORTANT NOTE:

If you have already run this connection for your system and want to change a few parameters, go to Changing an Input File. Be aware, if you have exited out of the project, even if you have saved your system and the related tables, *you MUST ALWAYS run Input Block A, before generating a new input file and running the model.*

Related Topics:

Changing an Input File

An Overview of Each Input Block

Executing a Model Run

Changing an Input File

Once you have initially established your input blocks (see Creating an Input File), it is possible to change just one block of the main input file. This ability is an advantage if you want to either correct a possible mistake or see the changes that may occur in the results, if a constant or a parameter changes (i.e. investigate model sensitivity).

To accomplish this task, first you must have run the entire Input File process at some point. It is not necessary to have run this process during the active session. You could have run it in an earlier session. If so, the free form text file created by Avenue (a.txt, b.txt, etc.) will still be present in your working directory. It will then only be necessary to rerun those blocks which you have changed information to.

Important things to remember:

- You must *ALWAYS* run Input Block A at the beginning of a session. This Input Block establishes your working directory, which is linked to the other subroutines that create Input Blocks B through J.
- The text files which are created and used by the formatting FORTRAN program are always named a.txt, b.txt, c.txt, etc. If you change an input block, but want preserve the first input file, be sure to give the new input file a different name, when prompted for a name in the input file generation However, with the project tables, if you change some parameters, unless you create an entirely new table in dBase or ArcInfo, your old information will not be preserved.
- If you change some parameters in the tables, be sure to rerun all input blocks which are affected by those characteristics you have changed. For example, if you change the cross sectional areas, it will be necessary to rerun both Input Block B: Exchanges, and Input Block C: Volumes, since the cross sectional area is used in both of these blocks. When in doubt, consult the WASP5 User's Manual B supplied with the program.

To change the an Input Block:

1. If the parameters to be changed are contained in a table, you must first edit the table. Make the table which you want to edit active and choose Table: Start Editing. Then, choose the editing icon from the tool bar and change the values. When complete, select Table: Stop Editing from the menu. Only dbf Tables can be edited. If you are trying to edit an INFO table, you must first export it as a dbf file, open back up in the project, and then rename it to the correct name. NOTE: To edit the main segment parameters table, you must first unjoin the table from the main segmentation table. To do this, have the main segmentation table active and choose Table: Remove All Joins. You should then run Input Block A to rejoin the tables.

2. Once you are done editing the table (if necessary), set the project window active and select Input File from the menu bar. A menu will appear showing you all ten input blocks. Choose the input block you wish to recreate and Avenue will regenerate the necessary text files.

3. When complete, choose BOD/DO Model: Generate Input File, to recreate the input file with the new information.

Related Topics

Creating an Input File

An Overview of Each Input Block

Executing a Model Run

Table of Contents

An Overview of Each Input Block

Input Block A: Model Identification and Simulation Control

This Input Block contains basic simulation information and model preferences . The script reads how many segments are present and queries the user to choose model preferences such as length of model run, preferred time step, and print intervals. This script also asks for a model title, which will be printed on the first line of the input file, and the input file name (*.inp). Be sure to use 8-3 convention when naming the input file and use the extension ".inp".

The following defaults are set in the generation of this input block:

* Presently, the connection set to handle just Simple Street-Phelps Modelling (BOD/DO Process)
* Backward differencing is always used.
* A transport file is always generated.
* The first six segments' solutions are those which are displayed on the screen, during a model run.
* The same maximum time step is used throughout the model run.
* The same print interval is used throughout the model run.

Input Block B: Exchange Coefficients

This Input Block describes the exchange coefficients for surface water (pore water exchanges are not set in the connection, as of yet). The script reads the lengths of the segments to calculate the characteristic length between segments. It also determines if the boundary segments are perpendicular to the main segmentation and compensates in the characteristic length, if it is. It also reads the exchange coefficients and cross-sectional areas for two neighbouring segments. It chooses the smaller area and the exchange coefficient for printing to the file. Finally, if a benthic sediment segment is set, the water column exchange between each overlying water segment and the pore water in the sediment segment are printed to the text file.

The following defaults are set:

* All exchange coefficients are steady-state
* All exchange coefficients are written to the text file in m^2/sec and lengths are in m.

Input Block C: Volumes

This Input Block describes the segment volumes for the system. It also sets the hydraulic geometry parameters for calculating segment depth and velocity. These geometry parameters are used to calculate reaeration (if necessary) or volitilization from the segments; they are not used in transport calculations. For this connection, a constant (in time and space) reaeration coefficient has been assumed -- therefore, these geometry parameters are not used to calculate reaeration in this particular set up. The script reads the cross-sectional areas (in m^2) and multiplies it by that segment's length (m) to obtain the volume. The script also asks the user to choose the following: 1) water column volume option, 2) benthic volume option, and 3) benthic time step.

The following defaults are set:

* Volumes are written to the text file in m^3
* Geometry parameters do not spatially vary
* Only the first benthic volume option is possible.

Input Block D: Flows

This Input Block provides the advective transport flows that are used in the model. Presently, only flows for WASP Flow Field 1 (Water Column) are used in the connection. First, the script asks the user to choose a flow option. These options are described below:

Flow Option 1: Field one flows are specified directly by the user. Individual flows at each segment interface are summed by the model, and the net flow us applied across the interface.

Flow Option 2: Field one flows are specified directly by the user. Individual flows at each segment interface are applied directly by the model.

Flow Option 3: Flows are read from a formatted file created by DYNHYD or other hydrodynamic model. If this option is chosen, the user will be asked to enter the file name of the text file which hold this information.

The script in ArcView assumes that each boundary condition given has an associated flow and each main segment has just one flow input. The script will track each flow input from its upper-most boundary to the most downstream segment. In order to do this process, each boundary segment and its corresponding upstream and downstream segments are read from the "Boundary Segments" Table. The downstream segment should be a main segment. The main segment is then found in the "Main Segmentation" Table, with its downstream segment. The script will continue to look for the successive downstream segments, until the most downstream main segment is reached. This flow route is printed to the text file and, at the end of the flow path, the actual flow from the original boundary segment is printed.

The table, "flow.dbf" is created during this input block determination, by doing a units conversion on the "Flow Accumulation Values" Table. This table will give the total cumulative flow, the incremental flow, the runoff (from the Runoff Accumulation Table) and the baseflow into each main segment in m^3/sec. These flows are joined to the "Main Segmentation" table and the incremental flows are used for Input Block D.

If "Dry Weather Conditions" is selected during the generation of this input blovk, only the baseflow will be printed to the text file as the flow in the system.

The following defaults are set:

* All flows are in m^3/sec. The percentages of the total flow are written to the file, along with the total flow value.

* The flow is steady state

* The number of flow fields is set to 1: Water column only (no pore water flows)

Input Block E: Boundaries

This Input Block describes the boundary segments and their concentrations. Model boundaries consist of those segments that import, export, or exchange water with locations outside the main network. A boundary is either a tributary inflow, a downstream outflow, or an open water end of the model network across which dispersive mixing can occur. The boundary concentrations are read from the "Boundary Segments" Table.

The following defaults are set:

* Since the model is set just for Simple Streeter-Phelps Model, only BOD and DO are considered in the boundary concentrations

* All concentrations are read and written in units of mg/L

* The boundary conditions are steady-state

Input Block F: Waste Loads

This Input Block writes the BOD and DO loads into each main network segment. The script does two things:

1. Reads the point source BOD loads into each segment and converts the value to kg/day
2. Calculates the non-point source BOD from the "BOD Loading Values" Table in kg/day

Since, presently, the model is set for steady state, the non-point BOD loads are added to the point source BOD loads to get a total load to the segment. Once the non-point source loads are set to time varying, a separate non-point source file will have to be generated.

This Input Block also generates a table called "load.dbf". This table gives the non-point source loads in kg/day for BOD. It will also give the total point source BOD loading in kg/day for each main segment. If "Dry Weather Conditions" is selected during Input Block D generation, only the point source loads are written to the text file. Non-point source loads will be ignored.

The following defaults are set:

* All loads are steady state

* Only BOD and Do are considered

Input Block G: Parameters

This input block reads the necessary parameters for the Level One EUTRO model. As the complexity level increases, the number of parameters needed will increase. Presently, only four parameters are needed:

1. Temperature, read from the segment tables in °C -- temperature is used to correct for deviations from the standard (20°C) and DO saturation.
2. Sediment oxygen demand (SOD), read from the tables in g/m^2-day
3. SOD theta correction, input by user -- used to correct SOD for temperatures deviating from 20°C.
4. Salinity, read from the tables -- used to calculated DO saturation

The following defaults are set:

* Temperature does not vary in time

* SOD theta does not vary in time or space

Input Block H: Constants

This Input Block queries the user for the necessary constants needed to run the Simple Streeter-Phelps model. The definition of the constants will vary, depending upon the structure and kinetics of the systems comprising each model. For the present model, only two constants are needed:

1. CBOD deoxygenation rate at 20°C, per day.
2. Reaeration rate constant at 20°C for entire water body, per day.

The following defaults are set:

* The constants do not vary in time or space

Input Block I: Time Functions

If the model were non-steady state for any parameter, this Input Block would use time functions to vary the specific parameter. Presently, none of the parameters are set to vary in time, so this block is default to 0.

Input Block J: Initial Conditions

This Input Block describes the initial conditions for each system in the model. Presently, only DO and BOD are considered in this script. The initial conditions are read from the segment tables. If the model is going to run until equilibrium is reached, the initial conditions will not affect the results.

NOTE: If time variant results are desired, additional programming should be done to set the initial conditions in the boundary segments to unique values. Presently, the initial conditions in the boundary segments are set to the boundary concentrations.

The following defaults are set:

* The dissolved fraction of BOD is set at 0.5

* The dissolved fraction of DO is always 1.0

* The maximum value for all systems is 1.0e8

* Solids Field 3 transports BOD is its particulate form

* Solids Field 5 transports DO

* All densities are set to 1.0 (EUTRO does not use those values)

* All initial conditions are in mg/L

Related Topics:

Creating an Input File

Changing an Input File

Table of Contents

Model Calibration

Before running the DO/BOD model in EUTRO, the system being modelled must first be calibrated. The user can perform this calibration without the help of the ArcView connection, or utilize the ArcView to assist in the model calibration.

For calibration, the TOXI5 is used to model a conservative substance in the system. Typically, at least two boundary conditions are set to a known concentration value of the substance; the rest of the segments are set to initial conditions of 200 ppt. The model is run and once equilibrium in the system is reached, the calculated values of the conservative substance can be compared to known values.

For the calibration in this connection, salinity is used as the conservative substance. To run the model calibration perform the following steps:

1. Have all tables and views necessary for a normal model run open and named correctly.
2. From the menu bar choose Model Calibration: Write Input Information. During this input generation you will be asked the number of boundary conditions and then asked to input the segment numbers which are the boundary conditions, one at a time.
3. When complete choose Model Calibration: Generate Input File.
4. Once the input file is done, choose Model Calibration: Run Model Calibration. This choice will execute TOXI5, when a list of files appears on the screen, choose the name of the input file you just created.
5. You can then review the results by processing and viewing the output.

As in the regular input file generation, the avenue program creates thirteen text files. A number of them are identical to the blocks needed for a BOD/DO model run. Specifically, input blocks B,C,D, and I are the same. For this reason, the same free form text files (b.txt,c.txt,d.txt, and i.txt) are written to the working directory. For all other input blocks, the free form text files have the prefix "cal" (i.e. cala.txt).

Related Topics:

Table of Contents

Executing a Model Run

Once an input file is created in ArcView, you can execute a model run by simply choosing BOD/DO Model: Run EUTRO5. Be sure that you have generated the input file before performing this step. When the model is executed, a DOS window will appear and the EUTRO interface will be shown. When a list of input files is shown, choose the input file you have created. Do not press the "ok" button on the message box until the model is completed in running. By pressing 'OK" you are informing ArcView that the model is done and it can now exit from DOS and return to the ArcView interface. NOTE: The EUTRO5 or TOXI5 models do not have to be executed through ArcView. It is possible to get ArcView to write numerous input files for these models and then run the model separately, through DOS. The final output can then be processed and viewed through ArcView.

Related Topics:

Table of Contents

Creating an Input File

Viewing the Output

There are three main steps to processing and viewing the output from a model run. Note that the file you want to process did not have to be created or executed in ArcView to process. But, if charts and coverages want to be created, the associated original tables and coverages to the model run should be present in ArcView. The three main steps are:

1. Checking the model parameters
2. Processing the output file
3. Viewing the output

All of these steps can be executed from either the BOD/DO Model menu or the Calibration Model menu. Be sure you choose the correct menu, corresponding to the model you are presently working with.

Step 1: Checking the model parameters

This item just reinitializes the model options and file information for ArcView to process the output. With this option, it is possible to create a number of output files with numerous model runs and then process them, one at a time, without having to go back and rerun the model. Be sure that you give the correct output file name for the model run you are interested in. If you are working with the calibration model, the output file will be named the same as the input file name you gave it, with a ".tdf" extension. The BOD/DO model with follow the same convention, with an ".edf" extension.

This step should ALWAYS be executed before processing and viewing an output file.

In addition, you should check the script, vwout, to be sure that the proper coverage and directory name is being referenced in the code (see Setting up the ArcView/WASP5 Connection for more information).

Step 2: Processing the output file

This option executes a FORTRAN program which will write a text file with either salinity, or BOD and DO measurements, for every segment at each time step. The text file will then be imported into ArcView as a dBase Table. You will be prompted to name each table, as it is processed. Be sure, if you are processing a number of output files, that you give the tables descriptive names so that they can be differentiated in the project.

Step 3: Viewing the output

The first thing you must do, after executing this command is to choose a table with which you want to work. Once selected, all charts or coverages created will be linked to that table. Although the table choice box will show you a list of all available tables, only those that were created by the above process step can be viewed using this menu option. If you choose a table that does not have the proper format, ArcView will exit you out of the view output script. Be sure that the table you select corresponds to the parameters you set in Step 1, above. If it does not, then go back and reset the parameters correctly. This need is because the output viewing steps (such as coverage creation) uses some of these parameters in order to execute some Avenue script commands.

Within this step, there are five options. There are as follows:

1. View the table you have chosen
2. Create a chart of concentration vs. time for a chosen segment
3. Create a chart of concentration vs. segment number for a chosen time
4. Create a coverage of the concentration at the last time step in the table
5. Create a "movie" of four coverages which display concentration at four chosen times

Option 1: Viewing the Table

By choosing this option, the table you have selected will open and become the active view. You can then view the output or create your own charts manually from this table, if desired.

Option 2: Concentration vs. Time

This option will open the view, "Segmentation" for you and prompt you to activate the "bug" icon and choose a segment. If, when the view opens, you do not see the bug icon, try resizing the ArcView window. The "bug" icon should be on the far right bottom tool bar. Once it is activated, be sure that the proper theme (either Main Segmentation or Boundary Segmentation) is highlighted and click on the segment you want to graph. You will then be asked a few chart options, including color and the name of the y-axis. Afterwards, a chart, showing concentration vs time (for all time steps) will appear. If you want just particular times plotted, you should manually go in and select those times on the linked table. The chart will then change to show the selected times, accordingly.

Option 3: Concentration vs. Segment Number

This option just prompts you for the time at which you want to display the concentration values. A bar chart showing the concentration for each segment (main and boundary) will then appear. If you want to see a different time, just select a different time on the linked table and the chart will change to reflect the new selection.

Option 4: Coverage at Final Time

This option will create an ArcView coverage of the concentration at the final time in the model. The script automatically brings the coverage up on the view and shows the concentration values for that time, in a ramped arc coverage (from blue to red).

Option 5: Movie of Four Chosen Times

This option allows you to choose four given times from all possible times in a chosen table. The script will then open a view and create a coverage, at a time delay chosen by the user. Each coverage will have eight intervals of concentration. The script will determine the min and max values within the time steps you have chosen and ramp the coverage from gray to blue in eight intervals. These intervals will stay constant for all four times, so that changes in concentrations can be viewed consistently. After the script is complete, the user will have four new coverages of concentration at each time step.

NOTE: All charts which you create are ALWAYS "linked" to a given table. If you make changes to that table or to you selections within the table, the chart will change, accordingly. For example, if you have created a chart of concentration vs. segment at time = 10 days. If you choose to create another chart of concentration vs. segment at time = 19 days, and link it to the same table, the first chart will also change. To avoid this problem, you can create or add multiple copies of the same table to the project and link a chart to each table. You can do this by either adding the dbf file numerous times and renaming the table so that it is more descriptive. Or, you can process the same output a number of times and just keep changing the table name when prompted for a name. Then, only link one chart to each table.

Related Topics:

Table of Contents

About WASP5

WASP5 stands for Water Quality Analysis Simulation Program Modelling System. It is developed by the Environmental Protection Agency (EPA) at the Center for Exposure Assessment Modelling (CEAM) in Athens, Georgia. The program consists of a main program, WASP, and three subprograms: EUTRO, TOXI, and DYNHYD. EUTRO is used to model BOD/DO and eutrophication; TOXI, toxic chemicals and model calibration; and DYNHYD, system hydrodynamics.

The program is available, shareware, from the EPA ftp site and on the World Wide Web. Although the current version of WASP (5.10), has a user interface entitled WISP (WASP Interactive Support Program), the ArcView connection discussed here does NOT utilize this interface. Due to present memory constraints, it is not possible to run WISP, while running the Windows environment necessary for ArcView. As a result, only the EUTRO and TOXI executables (with their related error and message files) are needed for this ArcView connection. The executables, along with the other necessary files, should be located in the working directory designated during Input Block A execution.

For questions or problems regarding model execution and parameters, the user should refer to the WASP User's Manual A and B, available with the model. Further questions regarding the model should be addressed to:

Center for Exposure Assessment Modelling
U.S. Environmental Protection Agency
Office of Research and Development
Environmental Research Laboratory
960 College Station Road
Athens, GA 30605-2720
706 543 3549

Related Topics:

Table of Contents

WASP v.5.1 September 1993

EPA ftp site: earth1.epa.gov

Directory: /pub/athens/

EPA www site: ftp://earth1.epa.gov/pub/athens/wwwhtml/wasp.htm

About this Connection

This connection was created as part of a research project at the Department of Civil Engineering at University of Texas at Austin. The project dealt with dissolved oxygen modelling of a ship channel, located off of Galveston Bay, Texas. Partial funding for this project was provided by the National Science Foundation.

The GIS/WASP connection is written in Avenue, under ArcView 2.1. The FORTRAN programs used to format the input files (outgen.exe: general input and calgen.exe: calibration input) and process the output (calout.exe and modout.exe) were compiled with Microsoft FORTRAN. The version of WASP which was connected to ArcView was 5.10 (WASP5).

If a CD ROM is supplied with this project, all coverages were generated using ArcInfo 7.03 and ArcView 2.1. To execute this program and view the example on the CD ROM, refer to Setting up the ArcView/WASP5 Connection in order to view the demo from the CD.

For a complete description on the methodology behind the coverage creation and model connection refer to:

Benaman, J. *Modeling of Dissolved Oxygen in the Houston Ship Channel, using WASP and Geographic Information Systems.* Master's Thesis. Department of Civil Engineering. The University of Texas at Austin. December 1996.

Related Topics:

Helpful References

Table of Contents

Limitations and Important Notes Concerning this Connection

Presently, this model connection has the following limitations:

- Only steady-state input files can be run
- The connection is best set up for a river system, or tidally influenced river system
- Simple Streeter-Phelps Model is the complexity level -- considers just Biochemical Oxygen Demand (BOD) and Dissolved Oxygen (DO)
- Only water column flow is considered in this connection. The connection is not presently set up for 2-layer water systems or sediment transport.
- Specific assumptions and limitations with each input block are discussed in An Overview of Each Input Block

Other Notes on this Connection:

- Input Block A must always be run at the beginning of each session in order to set the working directory.
- The working directory should be that which holds the model executables (eutro5.exe and toxi5.exe) and the FORTRAN formatting programs (outgen.exe, calgen.exe, calout.exe, modout.exe).
- Segment number "1" should be the first main segment in your model. The main segments should all be numbered first, followed then by the boundary segments.
- All unit conversions encrypted in the scripts are based on the units given in the tables descriptions. It is important to note that the tables generated from grid used a 100m x 100m cell-size. If different units are used in the tables, then the user must go into the scripts and change the unit conversions.
- Each time you run Input Block A, the main segment parameters table is joined to the main segmentation table. To avoid successive joining, be sure to unjoin the tables before rerunning Input Block A. Have the main segmentation table active and choose Table: Remove all Joins.
- Be sure that each time you process or view an output file, you check the parameters. This step is done by choosing "Check Model Parameters" under either BOD/DO Model or Calibration Model, depending on which out file you will be processing. This step tells ArcView what output file you want to process and reinitializes some of the model options (i.e. time step, print interval, etc) in order to read the output file correctly.
- All charts which you create are ALWAYS "linked" to a given table. If you make changes to that table or to you selections within the table, the chart will change, accordingly. For example, if you have

created a chart of concentration vs. segment at time = 10 days. If you choose to create another chart of concentration vs. segment at time = 19 days, and link it to the same table, the first chart will also change. To avoid this problem, you can create or add multiple copies of the same table to the project and link a chart to each table. You can do this by either adding the dbf file numerous times and renaming the table so that it is more descriptive. Or, you can process the same output a number of times and just keep changing the table name when prompted for a name. Then, only link one chart to each table.

<u>Related Topics:</u>

Table of Contents

Troubleshooting

Most errors will occur because the tables or fields are not named correctly. Be sure that the table and field names (or aliases) correspond to those outlined in the Tables Needed for Processing.

Other errors which may occur:

Error: Message Box reading A Nil Object does not recognize the request AsFileName.

Soln: You need to run Input Block A to set the working directory.

Error: The parameters written in the input file are not correct when spot checked.

Soln: The units in your tables may not correspond to the units necessary for the conversions set in the scripts. If they are different, you must either convert them and create a new table with units that agree to those outlined in the Tables Needed for Processing, or you must go into the scripts and change the unit conversions to match those needed for WASP.

Error: Message Box reading: Segmentation Violation!

Soln: For some unknown reason, occasionally, Avenue will print this error. It is unknown why, but usually it does not interfere with the input file generation. Just click the "OK" button and the program should continue without problems. It is smart, though, to save you project often, especially if you are making many changes.

Error: Repeated Fields in the "Main Segmentation" Table

Soln: Each time you run the entire input file, the calibration input file, or Input Block D, the table "flow.dbf" is joined to the "Main Segmentation" Table. Also, the Input Block A joins the Main Segmentation Table to its corresponding parameters table. To correct multiple fields, select the Main Segmentation Table and Choose Table: Remove all Joins from the menu bar. You would then have to rerun Input Blocks A and D (or all the input blocks at once) to rejoin the tables.

Error: "Bug" Icon used to choose a segment for chart creation is not shown on the View Tool Bar.

Soln: Resize the entire ArcView Window and the icon should appear on the second tool bar, on the far right.

Error: Error Box reading "AV Script Out of Range 0-1"

Soln: This message box probably appears when you are trying to select a segment from the view for creating a chart. Be sure that the "Main Segmentation" or "Boundary Segmentation" theme is active on the view and highlighted before clicking on the segment. If this error box occurs other than this instance, try saving your project, exiting ArcView, and then reopening ArcView and your project.

Error: The FORTRAN output formatting program does not execute or you get a message similar to "out of memory" or "stack overflow".

Soln: The maximum length of the output file to be processed is dependent on the amount of available memory of the computer running the connection. For EUTRO5, 18 lines of text are written for each segment at every time step. In the same way, 7 lines of text are written for each segment at each time step in the TOXI5 output file. If an "out of memory" or "stack overflow" error occurs when trying to process the output data file, the number of times steps written to the output file can be reduced to decrease the number of lines in the WASP5 output files.

Error : Error Box reading nil object segSrcName does not understand the command Make.

Soln: The script, *vwout*, is not accessing the correct directory and/or coverage name. Refer to Setting up the ArcView/WASP5 Connection on how to correct this problem.

Related Topics:

Table of Contents

Helpful References

Ambrose, R.B., T.A. Wool, and J.L. Martin. *The Water Quality Analysis Simulation Program, WASP5 Part A: Model Documentation.* Environmental Research Laboratory. Athens, Georgia. September 1993.

Ambrose, R.B., T.A. Wool, and J.L. Martin. *The Water Quality Analysis Simulation Program, WASP5 Part B: The WASP Input Data Set.* Environmental Research Laboratory. Athens, Georgia. September 1993.

Benaman, J. *Modelling Dissolved Oxygen in the Houston Ship Channel using WASP and Geographic Information Systems.* Master's Thesis. Department of Civil Engineering. University of Texas at Austin. December 1996.

Environmental Systems Research Institute, Inc. *Avenue: Customization and Application Development for ArcView.* Redlands, California. 1994.

Environmental Systems Research Institute, Inc. Understanding GIS: The ArcInfo Method. Redlands, California. 1995.

Related Topics:

Table of Contents

Appendix H
Houston Ship Channel Water Quality Data

		1978 Data			1982 Data			Modeled	1992 Avgs			1978 Data			1982 Data	
Station Name	Marker * (km)	Sal High (ppt)	Sal Low (ppt)	Avg Sal (ppt)	Sal High (ppt)	Sal Low (ppt)	Avg Sal (ppt)	Salinty (ppt)	Sal High (ppt)	Sal Low (ppt)	Avg Sal (ppt)	DO High (mg/L)	DO Low (mg/L)	Avg DO (mg/L)	DO High (mg/L)	DO Low (mg/L)
Buffalo Bayou @																
West Belt Drive	-42.20															
Shepherd Drive	-14.60	0.40	0.30	0.30	0.40	0.30	0.40					4.20	2.70	3.40	3.90	3.00
IH 45	-10.50				0.40	0.30	0.30								3.20	2.40
US 59	-7.50				0.40	0.20	0.30								4.00	0.80
Lockwood Drive	-4.00				1.00	0.20	0.40								3.70	0.70
69th Street	-1.60	2.90	2.20	2.60	6.40	0.50	3.80					0.00	0.00	0.00	1.90	0.10
HSC @																
Turning Basin	0.00	14.70	7.10	11.80	10.00	3.00	6.00					2.70	0.00	0.20	2.20	0.00
Wharf 20	1.50				9.40	3.80	6.80								1.30	0.00
Brays Bayou	2.90				10.00	1.90	7.00								1.90	0.00
IH 610	3.90	16.70	9.40	13.00	10.00	4.90	7.40					1.70	0.00	0.10	2.40	0.00
Sims Bayou	6.50	16.40	9.40	13.70	10.60	3.20	7.40					1.70	0.00	0.20	2.20	0.00
Vince Bayou	8.50				11.90	6.20	8.20								2.00	0.00
Washburn Tunnel	9.70	17.10	10.90	13.70	13.00	7.70	9.40					2.10	0.00	0.30	2.30	0.00
Hunting Bayou	11.50	13.60	11.30	12.40	13.60	7.40	9.50					1.40	0.00	0.40	2.20	0.00
Greens Bayou	14.80	10.20	14.30	12.90	14.00	7.70	10.00					2.10	0.00	0.50	2.30	0.00
Beltway 8	17.40				14.30	8.50	10.80								2.40	0.00
Patrick Bayou	20.80	14.70	14.00	14.30	18.80	9.70	11.90					7.10	0.30	0.70	1.90	0.00
Carpenter Bayou	23.20	15.40	14.00	14.80	16.00	10.00	11.40					3.20	0.60	1.50	2.50	0.40
Lynchburg Ferry	25.00				19.40	10.60	12.50								4.70	1.30
10	-4.00							3.90								
1	1.60							6.90								
2	5.00							7.13								
3	7.80							7.41								
4	10.10							7.70								
5	13.40							8.17								
6	17.30							8.81								
7	20.40							9.55								
8	23.30							10.10								
17	26.60							10.70								
H12	26.00								15.70	6.10	10.90					
H13	24.00								15.06	3.26	9.16					
H14	22.00								15.45	4.25	9.85					
H15	17.00								14.15	4.75	9.45					
H16	13.00								14.05	5.76	9.96					
H17	8.00								14.65	3.07	7.87					
H18	2.00								13.25	2.32	5.72					
H19	0.00								14.65	1.65	6.45					
H20	-4.00								12.85	0.00	1.82					

* 0 = Turning Basin

Sources: 1978 and 1982 Data from (TDWR, 1984) 1971 Data from (Espey, *et al.* , 1971)
1992 Data from (Ward and Armstrong, 1992)

			Modelled		1992 Avgs		1978	1982	Modelled		1992 Avgs		1978	1982	1971
Station Name	Marker * (km)	Avg DO (mg/L)	DO (mg/L)	DO High (mg/L)	DO Low (mg/L)	Avg DO (mg/L)	BOD (mg/L)	BOD (mg/L)	BOD (mg/L)	BOD High (ppt)	BOD Low (ppt)	Avg BOD (ppt)	Flow (cms)	Flow (cms)	Flow (cms)
Buffalo Bayou @															
West Belt Drive	-42.20												1.67		
Shepherd Drive	-14.60	3.70					2.00	2.00						3.28	
IH 45	-10.50	3.10						1.00							
US 59	-7.50	1.50						1.50							
Lockwood Drive	-4.00	1.90						4.00							
69th Street	-1.60	0.30					10.00	5.00							
HSC @															
Turning Basin	0.00	0.20					5.00	1.50							
Wharf 20	1.50	0.20						1.50							
Brays Bayou	2.90	0.10						1.00							4.67
IH 610	3.90	0.20					2.00	1.00							
Sims Bayou	6.50	0.30					4.00	1.00							1.25
Vince Bayou	8.50	0.40						1.00							2.38
Washburn Tunnel	9.70	0.30					4.00	1.00							
Hunting Bayou	11.50	0.40					2.00	1.00							0.99
Greens Bayou	14.80	0.60					2.00	1.00							3.37
Beltway 8	17.40	0.40						1.00							
Patrick Bayou	20.80	0.70					2.00	1.00							
Carpenter Bayou	23.20	1.30					2.00	1.00							0.99
Lynchburg Ferry	25.00	2.40						1.50							
10	-4.00		2.84						2.83						15.83
1	1.60		2.49						5.38						15.83
2	5.00		2.53						5.39						20.5
3	7.80		2.55						5.44						21.75
4	10.10		2.59						5.51						24.13
5	13.40		2.68						5.65						25.12
6	17.30		2.88						5.97						28.49
7	20.40		3.07						6.34						28.49
8	23.30		3.23						6.67						29.48
17	26.60		3.51						7.18						71.98
H12	26.00			6.04	1.24	3.64				12.12	2.72	7.42			
H13	24.00			8.11	1.91	5.01				8.39	0.59	4.49			
H14	22.00			3.44	-0.16	1.64				10.26	-0.14	5.06			
H15	17.00			4.35	0.15	2.25				6.53	0.53	3.53			
H16	13.00			1.78	-0.42	0.68				11.45	1.05	6.25			
H17	8.00			3.71	-0.09	1.81				9.24	0.84	5.04			
H18	2.00			3.86	-1.14	1.36	280			11.08	3.28	7.18			
H19	0.00			3.37	-0.63	1.37				13.87	0.47	7.17			
H20	-4.00			5.63	0.43	3.03				17.34	-1.06	8.14			

* 0 = Turning Basin

Sources: 1978 and 1982 Data from (TDWR, 1984) 1971 Data from (Espey, *et al.*, 1971)
1992 Data from (Ward and Armstrong, 1992)

Appendix I
List of Acronyms and Nomenclature

Acronyms

aat	arc attribute table
AGNPS	Agricultural Non-Point Source
aml	ArcInfo macro language
ARS	Agricultural Research Service
BOD	biochemical oxygen demand
CBOD	carbonaceous biochemical oxygen demand
CEAM	Center for Exposure Assessment Modeling
CSO	combined sewer overflow
dbf	dBase file
DEM	digital elevation model
DLG	digital line graph
DO	dissolved oxygen
DYNHYD5	WASP5's hydrodynmics subprogram
EH&A	Espey, Huston, and Associates
EPIC	Erosion Productivity Impact Calculator
EROS	Earth Resources Observation System
ESRI	Environmental Systems Research, Inc.
EUTRO4	WASP4's eutrophication subprogram
EUTRO5	WASP5's eutrophication subprogram
ftp	file transfer protocol
GBNEP	Galveston Bay National Estuary Program
GEO-WAMS	Geographically-based Watershed Analysis and Modeling System
GIS	Geographic Information Systems
GLEAMS	Ground Water Loading Effects of Agricultural Management Systems
GRASS	Geographic Resource Analysis System
GRS80	Global Reference System Spheroid 1980
HSC	Houston Ship Channel
HSPF	Hydrologic Simulation Progran FORTRAN
ILWIS	Integrated Land and Water Information System
IWMM	Integrated Watershed Management Model
MICRO-FEM	A European groundwater model
MODFLOW	US Geological Service groundwater model

Acronyms (cont.)

NaCl	Sodium Chloride (salt)
NAD83 (27)	North American Datum 1983 (1927)
NPS	non-point source
pat	polygon or point attribute table (depends on the type of coverage)
ppt	parts per thousand
SOD	sediment oxygen demand
SWRRBWQ	A Basin Scale Simulation Model for Soil and Water Resources Management
TDWR	Texas Department of Water Resources
TNRCC	Texas Natural Resource Conservation Commision
TOXI5	WASP5's toxic chemical subprogram
TWC	Texas Water Commision
USEPA	United States Enviornmental Protection Agency
USGS	United States Geological Survey
USGS-Albers	US Geological Service - Albers Equal Area Map Projection
vat	value attribute table
WASP	Water Quality Analysis Simulation Program
WASP4	WASP program, version 4 (1983)
WASP5	WASP program, version 5.1 (1993)
WGS84 (72)	World Geodetic System Datum 1984 (1972)

Nomenclature

a,b,c,and d	emperical coefficients or exponents
A_{ij}	interfacial area shared by segments "i" and j" (m^2)
C_5	concentration of carbonaceous biochemical oxygen demand (mg/L) (interpreted as total BOD for level one in EUTRO5)
C_6	concentration of dissolved oxygen (mg/L)
C_{bik}	concentration in boundary segment, "i" (mg/L)
C'_{bik}	adjusted concentration for boundary segment "i" (mg/L)
C_{ik}, C_{jk}	concentration of chemical "k" in segments "i" and "j" (mg/L)
C_s	dissolved oxygen saturation (mg/L)
D	depth of the overlying water column (m)

Nomenclature (cont.)

$E_{ij}(t)$	dispersion coefficient time function for exchange "ij" (m^2/day)
f_{DS}	fraction of dissolved CBOD
k_2	reaeration rate (/day)
K_{BOD}	half saturation constant for oxygen limitation (mg O_2/L)
k_d	deoxygenation rate @ 20 °C (/day)
L_{cij}	characteristic mixing length between segments "i" and "j" (m)
M_{ik}	mass of chemical "k" in segment I (g)
Q	channel flow (m^3/sec)
Q_{0i}	upstream inflow into boundary segment, "i" (m^3/day)
Q_{bf}	steady state baseflow upstream of segment "i" (m^3/day)
Q_{tot}	total flow upstream of segment "i"
$R\ x_{available}$	average yearly runoff depth for partially gauged station, averaged over the record available (mm/yr)
$R\ x_{1961-1990}$	average yearly runoff depth for a given gauge, x, adjusted to represent the entire period, 1961 to 1990 (mm/yr)
$R\ y_{available}$	average yearly runoff depth of four fully gauged stations, averaged over the record available for gauge x (mm/yr)
$R\ y_{1961-1990}$	average yearly runoff depth of for fully gauged stations, averaged over the entire period of record, 1961 - 1990 (mm/yr)
S_{bik}	boundary loading rate response of chemical "k" in segment, "i" (g/m^3-day)
SOD	sediment oxygen demand @ 20 °C (g/m^2-day)
T	temperature (°C)
V	channel velocity (m/sec)
V_i	volume of segment i (m^3)
v_{s3}	organic matter settling velocity (m/day)
Θ_D	deoxygenation temperature coefficient (--)
Θ_S	temperature coefficient (--)