

SWAT

Generated by Doxygen 1.8.16



<b>1 SWAT</b>	<b>1</b>
<b>2 Modules Index</b>	<b>7</b>
2.1 Modules List . . . . .	7
<b>3 File Index</b>	<b>9</b>
3.1 File List . . . . .	9
<b>4 Module Documentation</b>	<b>15</b>
4.1 parm Module Reference . . . . .	15
4.1.1 Detailed Description . . . . .	98
4.1.2 Variable Documentation . . . . .	99
4.1.2.1 igropt . . . . .	99
<b>5 File Documentation</b>	<b>101</b>
5.1 addh.f90 File Reference . . . . .	101
5.1.1 Detailed Description . . . . .	101
5.1.2 Function/Subroutine Documentation . . . . .	101
5.1.2.1 addh() . . . . .	101
5.2 albedo.f90 File Reference . . . . .	102
5.2.1 Detailed Description . . . . .	102
5.2.2 Function/Subroutine Documentation . . . . .	102
5.2.2.1 albedo() . . . . .	102
5.3 allocate_parms.f90 File Reference . . . . .	102
5.3.1 Detailed Description . . . . .	102
5.4 alph.f90 File Reference . . . . .	103
5.4.1 Detailed Description . . . . .	103
5.4.2 Function/Subroutine Documentation . . . . .	103
5.4.2.1 alph() . . . . .	103
5.5 anfert.f90 File Reference . . . . .	103
5.5.1 Detailed Description . . . . .	104
5.5.2 Function/Subroutine Documentation . . . . .	104
5.5.2.1 anfert() . . . . .	104
5.6 apex_day.f90 File Reference . . . . .	104
5.6.1 Detailed Description . . . . .	104
5.6.2 Function/Subroutine Documentation . . . . .	104
5.6.2.1 apex_day() . . . . .	105
5.7 apply.f90 File Reference . . . . .	105
5.7.1 Detailed Description . . . . .	105
5.7.2 Function/Subroutine Documentation . . . . .	105
5.7.2.1 apply() . . . . .	105
5.8 ascrv.f90 File Reference . . . . .	106
5.8.1 Detailed Description . . . . .	106
5.8.2 Function/Subroutine Documentation . . . . .	106

---

5.8.2.1 ascrv()	106
5.9 atri.f90 File Reference	107
5.9.1 Detailed Description	107
5.9.2 Function/Subroutine Documentation	107
5.9.2.1 atri()	107
5.10 aunif.f90 File Reference	107
5.10.1 Detailed Description	108
5.10.2 Function/Subroutine Documentation	108
5.10.2.1 aunif()	108
5.11 autoirr.f90 File Reference	108
5.11.1 Detailed Description	108
5.11.2 Function/Subroutine Documentation	109
5.11.2.1 autoirr()	109
5.12 bacteria.f90 File Reference	109
5.12.1 Detailed Description	109
5.12.2 Function/Subroutine Documentation	109
5.12.2.1 bacteria()	109
5.13 biozone.f90 File Reference	110
5.13.1 Detailed Description	110
5.13.2 Function/Subroutine Documentation	110
5.13.2.1 biozone()	110
5.14 bmp_det_pond.f90 File Reference	110
5.14.1 Detailed Description	110
5.14.2 Function/Subroutine Documentation	111
5.14.2.1 bmp_det_pond()	111
5.15 bmp_ri_pond.f90 File Reference	111
5.15.1 Detailed Description	111
5.16 bmp_sand_filter.f90 File Reference	111
5.16.1 Detailed Description	112
5.17 bmp_sed_pond.f90 File Reference	112
5.17.1 Detailed Description	112
5.18 bmp_wet_pond.f90 File Reference	112
5.18.1 Detailed Description	112
5.18.2 Function/Subroutine Documentation	112
5.18.2.1 bmp_wet_pond()	112
5.18.2.2 pipe_discharge()	113
5.19 bmpinit.f90 File Reference	113
5.19.1 Detailed Description	113
5.19.2 Function/Subroutine Documentation	113
5.19.2.1 bmpinit()	113
5.20 buffer.f90 File Reference	114
5.20.1 Detailed Description	114

---

5.20.2 Function/Subroutine Documentation . . . . .	114
5.20.2.1 buffer() . . . . .	114
5.21 burnop.f90 File Reference . . . . .	114
5.21.1 Detailed Description . . . . .	115
5.21.2 Function/Subroutine Documentation . . . . .	115
5.21.2.1 burnop() . . . . .	115
5.22 canopyint.f90 File Reference . . . . .	115
5.22.1 Detailed Description . . . . .	115
5.22.2 Function/Subroutine Documentation . . . . .	115
5.22.2.1 canopyint() . . . . .	115
5.23 caps.f90 File Reference . . . . .	116
5.23.1 Detailed Description . . . . .	116
5.23.2 Function/Subroutine Documentation . . . . .	116
5.23.2.1 caps() . . . . .	116
5.24 carbon_new.f90 File Reference . . . . .	116
5.24.1 Detailed Description . . . . .	117
5.24.2 Function/Subroutine Documentation . . . . .	117
5.24.2.1 carbon() . . . . .	117
5.25 carbon_zhang2.f90 File Reference . . . . .	117
5.25.1 Detailed Description . . . . .	118
5.25.2 Function/Subroutine Documentation . . . . .	118
5.25.2.1 carbon_zhang2() . . . . .	118
5.26 cfactor.f90 File Reference . . . . .	118
5.26.1 Detailed Description . . . . .	118
5.26.2 Function/Subroutine Documentation . . . . .	118
5.26.2.1 cfactor() . . . . .	118
5.27 clgen.f90 File Reference . . . . .	119
5.27.1 Detailed Description . . . . .	119
5.27.2 Function/Subroutine Documentation . . . . .	119
5.27.2.1 clgen() . . . . .	119
5.28 clicon.f90 File Reference . . . . .	119
5.28.1 Detailed Description . . . . .	120
5.28.2 Function/Subroutine Documentation . . . . .	120
5.28.2.1 clicon() . . . . .	120
5.29 command.f90 File Reference . . . . .	120
5.29.1 Detailed Description . . . . .	120
5.29.2 Function/Subroutine Documentation . . . . .	120
5.29.2.1 command() . . . . .	121
5.30 conapply.f90 File Reference . . . . .	121
5.30.1 Detailed Description . . . . .	121
5.30.2 Function/Subroutine Documentation . . . . .	121
5.30.2.1 conapply() . . . . .	121

5.31 confert.f90 File Reference . . . . .	122
5.31.1 Detailed Description . . . . .	122
5.31.2 Function/Subroutine Documentation . . . . .	122
5.31.2.1 confert() . . . . .	122
5.32 crackflow.f90 File Reference . . . . .	122
5.32.1 Detailed Description . . . . .	122
5.32.2 Function/Subroutine Documentation . . . . .	123
5.32.2.1 crackflow() . . . . .	123
5.33 crackvol.f90 File Reference . . . . .	123
5.33.1 Detailed Description . . . . .	123
5.33.2 Function/Subroutine Documentation . . . . .	123
5.33.2.1 crackvol() . . . . .	123
5.34 curno.f90 File Reference . . . . .	124
5.34.1 Detailed Description . . . . .	124
5.34.2 Function/Subroutine Documentation . . . . .	124
5.34.2.1 curno() . . . . .	124
5.35 dailycn.f90 File Reference . . . . .	124
5.35.1 Detailed Description . . . . .	125
5.35.2 Function/Subroutine Documentation . . . . .	125
5.35.2.1 dailycn() . . . . .	125
5.36 decay.f90 File Reference . . . . .	125
5.36.1 Detailed Description . . . . .	125
5.36.2 Function/Subroutine Documentation . . . . .	125
5.36.2.1 decay() . . . . .	125
5.37 depstor.f90 File Reference . . . . .	126
5.37.1 Detailed Description . . . . .	126
5.37.2 Function/Subroutine Documentation . . . . .	126
5.37.2.1 depstor() . . . . .	126
5.38 distributed_bmps.f90 File Reference . . . . .	126
5.38.1 Detailed Description . . . . .	127
5.39 dormant.f90 File Reference . . . . .	127
5.39.1 Detailed Description . . . . .	127
5.39.2 Function/Subroutine Documentation . . . . .	127
5.39.2.1 dormant() . . . . .	127
5.40 drains.f90 File Reference . . . . .	127
5.40.1 Detailed Description . . . . .	128
5.40.2 Function/Subroutine Documentation . . . . .	128
5.40.2.1 drains() . . . . .	128
5.41 dstn1.f90 File Reference . . . . .	128
5.41.1 Detailed Description . . . . .	128
5.41.2 Function/Subroutine Documentation . . . . .	128
5.41.2.1 dstn1() . . . . .	129

5.42 ee.f90 File Reference . . . . .	130
5.42.1 Detailed Description . . . . .	130
5.42.2 Function/Subroutine Documentation . . . . .	130
5.42.2.1 ee() . . . . .	130
5.43 eusle.f90 File Reference . . . . .	131
5.43.1 Detailed Description . . . . .	131
5.44 enrsb.f90 File Reference . . . . .	131
5.44.1 Detailed Description . . . . .	131
5.44.2 Function/Subroutine Documentation . . . . .	131
5.44.2.1 enrsb() . . . . .	131
5.45 estimate_ksat.f90 File Reference . . . . .	132
5.45.1 Detailed Description . . . . .	132
5.45.2 Function/Subroutine Documentation . . . . .	132
5.45.2.1 estimate_ksat() . . . . .	132
5.46 etact.f90 File Reference . . . . .	133
5.46.1 Detailed Description . . . . .	133
5.47 etpot.f90 File Reference . . . . .	133
5.47.1 Detailed Description . . . . .	133
5.47.2 Function/Subroutine Documentation . . . . .	133
5.47.2.1 etpot() . . . . .	133
5.48 expo.f90 File Reference . . . . .	134
5.48.1 Detailed Description . . . . .	134
5.48.2 Function/Subroutine Documentation . . . . .	134
5.48.2.1 expo() . . . . .	134
5.49 fcgd.f90 File Reference . . . . .	134
5.49.1 Detailed Description . . . . .	135
5.50 fert.f90 File Reference . . . . .	135
5.50.1 Detailed Description . . . . .	135
5.50.2 Function/Subroutine Documentation . . . . .	135
5.50.2.1 fert() . . . . .	135
5.51 filter.f90 File Reference . . . . .	135
5.51.1 Detailed Description . . . . .	136
5.51.2 Function/Subroutine Documentation . . . . .	136
5.51.2.1 filter() . . . . .	136
5.52 filtw.f90 File Reference . . . . .	136
5.52.1 Detailed Description . . . . .	136
5.52.2 Function/Subroutine Documentation . . . . .	136
5.52.2.1 filtw() . . . . .	136
5.53 finalbal.f90 File Reference . . . . .	137
5.53.1 Detailed Description . . . . .	137
5.54 gcycl.f90 File Reference . . . . .	137
5.54.1 Detailed Description . . . . .	137

5.55 getallo.f90 File Reference	137
5.55.1 Detailed Description	138
5.56 grass_wway.f90 File Reference	138
5.56.1 Detailed Description	138
5.56.2 Function/Subroutine Documentation	138
5.56.2.1 grass_wway()	138
5.57 graze.f90 File Reference	138
5.57.1 Detailed Description	139
5.57.2 Function/Subroutine Documentation	139
5.57.2.1 graze()	139
5.58 grow.f90 File Reference	139
5.58.1 Detailed Description	139
5.58.2 Function/Subroutine Documentation	139
5.58.2.1 grow()	139
5.59 gw_no3.f90 File Reference	140
5.59.1 Detailed Description	140
5.59.2 Function/Subroutine Documentation	140
5.59.2.1 gw_no3()	140
5.60 gwmod.f90 File Reference	140
5.60.1 Detailed Description	141
5.60.2 Function/Subroutine Documentation	141
5.60.2.1 gwmod()	141
5.61 gwmod_deep.f90 File Reference	141
5.61.1 Detailed Description	141
5.61.2 Function/Subroutine Documentation	141
5.61.2.1 gwmod_deep()	141
5.62 gwnutr.f90 File Reference	142
5.62.1 Detailed Description	142
5.62.2 Function/Subroutine Documentation	142
5.62.2.1 gwnutr()	142
5.63 h2omgt_init.f90 File Reference	142
5.63.1 Detailed Description	143
5.64 harvestop.f90 File Reference	143
5.64.1 Detailed Description	143
5.64.2 Function/Subroutine Documentation	143
5.64.2.1 harvestop()	143
5.65 harvkillop.f90 File Reference	143
5.65.1 Detailed Description	144
5.65.2 Function/Subroutine Documentation	144
5.65.2.1 harvkillop()	144
5.66 headout.f90 File Reference	144
5.66.1 Detailed Description	144



5.67 hhnoqual.f90 File Reference . . . . .	144
5.67.1 Detailed Description . . . . .	145
5.67.2 Function/Subroutine Documentation . . . . .	145
5.67.2.1 hhnoqual() . . . . .	145
5.68 hhwatqual.f90 File Reference . . . . .	145
5.68.1 Detailed Description . . . . .	145
5.68.2 Function/Subroutine Documentation . . . . .	145
5.68.2.1 hhwatqual() . . . . .	145
5.69 hmeas.f90 File Reference . . . . .	146
5.69.1 Detailed Description . . . . .	146
5.70 HQDAV.f90 File Reference . . . . .	146
5.70.1 Detailed Description . . . . .	146
5.71 hruaa.f90 File Reference . . . . .	146
5.71.1 Detailed Description . . . . .	147
5.71.2 Function/Subroutine Documentation . . . . .	147
5.71.2.1 hruaa() . . . . .	147
5.72 hruallo.f90 File Reference . . . . .	147
5.72.1 Detailed Description . . . . .	147
5.73 hruday.f90 File Reference . . . . .	147
5.73.1 Detailed Description . . . . .	148
5.73.2 Function/Subroutine Documentation . . . . .	148
5.73.2.1 hruday() . . . . .	148
5.74 hrumon.f90 File Reference . . . . .	148
5.74.1 Detailed Description . . . . .	148
5.75 hrupond.f90 File Reference . . . . .	148
5.75.1 Detailed Description . . . . .	149
5.75.2 Function/Subroutine Documentation . . . . .	149
5.75.2.1 hrupond() . . . . .	149
5.76 hrupondhr.f90 File Reference . . . . .	149
5.76.1 Detailed Description . . . . .	149
5.76.2 Function/Subroutine Documentation . . . . .	149
5.76.2.1 hrupondhr() . . . . .	149
5.77 hruyr.f90 File Reference . . . . .	150
5.77.1 Detailed Description . . . . .	150
5.78 hydroinit.f90 File Reference . . . . .	150
5.78.1 Detailed Description . . . . .	150
5.79 icl.f90 File Reference . . . . .	150
5.79.1 Detailed Description . . . . .	151
5.79.2 Function/Subroutine Documentation . . . . .	151
5.79.2.1 icl() . . . . .	151
5.80 impnd_init.f90 File Reference . . . . .	151
5.80.1 Detailed Description . . . . .	151

5.81 impndaa.f90 File Reference	151
5.81.1 Detailed Description	152
5.81.2 Function/Subroutine Documentation	152
5.81.2.1 impndaa()	152
5.82 impndday.f90 File Reference	152
5.82.1 Detailed Description	152
5.82.2 Function/Subroutine Documentation	152
5.82.2.1 impndday()	152
5.83 impndmon.f90 File Reference	153
5.83.1 Detailed Description	153
5.84 impndyr.f90 File Reference	153
5.84.1 Detailed Description	153
5.85 irr_rch.f90 File Reference	153
5.85.1 Detailed Description	154
5.85.2 Function/Subroutine Documentation	154
5.85.2.1 irr_rch()	154
5.86 irr_res.f90 File Reference	154
5.86.1 Detailed Description	154
5.86.2 Function/Subroutine Documentation	154
5.86.2.1 irr_res()	154
5.87 irrigate.f90 File Reference	155
5.87.1 Detailed Description	155
5.87.2 Function/Subroutine Documentation	155
5.87.2.1 irrigate()	155
5.88 irrsb.f90 File Reference	155
5.88.1 Detailed Description	156
5.88.2 Function/Subroutine Documentation	156
5.88.2.1 irrsb()	156
5.89 jdt.f90 File Reference	156
5.89.1 Detailed Description	156
5.89.2 Function/Subroutine Documentation	156
5.89.2.1 jdt()	157
5.90 killop.f90 File Reference	158
5.90.1 Detailed Description	158
5.90.2 Function/Subroutine Documentation	158
5.90.2.1 killop()	158
5.91 lakeq.f90 File Reference	158
5.91.1 Detailed Description	159
5.91.2 Function/Subroutine Documentation	159
5.91.2.1 lakeq()	159
5.92 latsed.f90 File Reference	159
5.92.1 Detailed Description	159

5.92.2 Function/Subroutine Documentation . . . . .	159
5.92.2.1 latsed() . . . . .	159
5.93 layersplit.f90 File Reference . . . . .	160
5.93.1 Detailed Description . . . . .	160
5.94 lid_cistern.f90 File Reference . . . . .	160
5.94.1 Detailed Description . . . . .	160
5.94.2 Function/Subroutine Documentation . . . . .	160
5.94.2.1 lid_cistern() . . . . .	160
5.95 lid_greenroof.f90 File Reference . . . . .	161
5.95.1 Detailed Description . . . . .	161
5.95.2 Function/Subroutine Documentation . . . . .	161
5.95.2.1 lid_greenroof() . . . . .	161
5.96 lid_porpavement.f90 File Reference . . . . .	162
5.96.1 Detailed Description . . . . .	162
5.96.2 Function/Subroutine Documentation . . . . .	162
5.96.2.1 lid_porpavement() . . . . .	162
5.97 lid_raingarden.f90 File Reference . . . . .	162
5.97.1 Detailed Description . . . . .	163
5.97.2 Function/Subroutine Documentation . . . . .	163
5.97.2.1 lid_raingarden() . . . . .	163
5.98 lidinit.f90 File Reference . . . . .	163
5.98.1 Detailed Description . . . . .	163
5.98.2 Function/Subroutine Documentation . . . . .	164
5.98.2.1 lidinit() . . . . .	164
5.99 lids.f90 File Reference . . . . .	164
5.99.1 Detailed Description . . . . .	164
5.99.2 Function/Subroutine Documentation . . . . .	164
5.99.2.1 lids() . . . . .	164
5.100 log_normal.f90 File Reference . . . . .	165
5.100.1 Detailed Description . . . . .	165
5.100.2 Function/Subroutine Documentation . . . . .	165
5.100.2.1 log_normal() . . . . .	165
5.101 lwqdef.f90 File Reference . . . . .	166
5.101.1 Detailed Description . . . . .	166
5.101.2 Function/Subroutine Documentation . . . . .	166
5.101.2.1 lwqdef() . . . . .	166
5.102 main.f90 File Reference . . . . .	166
5.102.1 Detailed Description . . . . .	166
5.103 modparm.f90 File Reference . . . . .	167
5.103.1 Detailed Description . . . . .	250
5.104 ndenit.f90 File Reference . . . . .	250
5.104.1 Detailed Description . . . . .	250

5.104.2 Function/Subroutine Documentation	251
5.104.2.1 ndenit()	251
5.105 newtllmix.f90 File Reference	251
5.105.1 Detailed Description	251
5.105.2 Function/Subroutine Documentation	251
5.105.2.1 newtllmix()	252
5.106 nfix.f90 File Reference	252
5.106.1 Detailed Description	252
5.106.2 Function/Subroutine Documentation	252
5.106.2.1 nfix()	252
5.107 nitvol.f90 File Reference	253
5.107.1 Detailed Description	253
5.107.2 Function/Subroutine Documentation	253
5.107.2.1 nitvol()	253
5.108 nlch.f90 File Reference	253
5.108.1 Detailed Description	253
5.108.2 Function/Subroutine Documentation	254
5.108.2.1 nlch()	254
5.109 nminrl.f90 File Reference	254
5.109.1 Detailed Description	254
5.109.2 Function/Subroutine Documentation	254
5.109.2.1 nminrl()	254
5.110 noqual.f90 File Reference	255
5.110.1 Detailed Description	255
5.110.2 Function/Subroutine Documentation	255
5.110.2.1 noqual()	255
5.111 npup.f90 File Reference	255
5.111.1 Detailed Description	256
5.111.2 Function/Subroutine Documentation	256
5.111.2.1 npup()	256
5.112 nrain.f90 File Reference	256
5.112.1 Detailed Description	256
5.112.2 Function/Subroutine Documentation	256
5.112.2.1 nrain()	256
5.113 nup.f90 File Reference	257
5.113.1 Detailed Description	257
5.113.2 Function/Subroutine Documentation	257
5.113.2.1 nup()	257
5.114 nuts.f90 File Reference	257
5.114.1 Detailed Description	258
5.114.2 Function/Subroutine Documentation	258
5.114.2.1 nuts()	258

5.115 openwth.f90 File Reference . . . . .	258
5.115.1 Detailed Description . . . . .	258
5.116 operatn.f90 File Reference . . . . .	259
5.116.1 Detailed Description . . . . .	259
5.116.2 Function/Subroutine Documentation . . . . .	259
5.116.2.1 operatn() . . . . .	259
5.117 orgn.f90 File Reference . . . . .	259
5.117.1 Detailed Description . . . . .	259
5.117.2 Function/Subroutine Documentation . . . . .	260
5.117.2.1 orgn() . . . . .	260
5.118 orgncswat.f90 File Reference . . . . .	260
5.118.1 Detailed Description . . . . .	260
5.118.2 Function/Subroutine Documentation . . . . .	260
5.118.2.1 orgncswat() . . . . .	260
5.119 orgncswat2.f90 File Reference . . . . .	261
5.119.1 Detailed Description . . . . .	261
5.119.2 Function/Subroutine Documentation . . . . .	261
5.119.2.1 orgncswat2() . . . . .	261
5.120 origtile.f90 File Reference . . . . .	262
5.120.1 Detailed Description . . . . .	262
5.120.2 Function/Subroutine Documentation . . . . .	262
5.120.2.1 origtile() . . . . .	262
5.121 ovr_sed.f90 File Reference . . . . .	262
5.121.1 Detailed Description . . . . .	262
5.121.2 Function/Subroutine Documentation . . . . .	263
5.121.2.1 ovr_sed() . . . . .	263
5.122 oxygen_saturation.f90 File Reference . . . . .	263
5.122.1 Detailed Description . . . . .	263
5.122.2 Function/Subroutine Documentation . . . . .	263
5.122.2.1 oxygen_saturation() . . . . .	263
5.123 percmacro.f90 File Reference . . . . .	264
5.123.1 Detailed Description . . . . .	264
5.123.2 Function/Subroutine Documentation . . . . .	264
5.123.2.1 percmacro() . . . . .	264
5.124 percmain.f90 File Reference . . . . .	264
5.124.1 Detailed Description . . . . .	265
5.124.2 Function/Subroutine Documentation . . . . .	265
5.124.2.1 percmain() . . . . .	265
5.125 percmicro.f90 File Reference . . . . .	265
5.125.1 Detailed Description . . . . .	265
5.125.2 Function/Subroutine Documentation . . . . .	265
5.125.2.1 percmicro() . . . . .	266

5.126 pestlch.f90 File Reference	267
5.126.1 Detailed Description	267
5.126.2 Function/Subroutine Documentation	267
5.126.2.1 pestlch()	267
5.127 pestw.f90 File Reference	267
5.127.1 Detailed Description	268
5.128 pesty.f90 File Reference	268
5.128.1 Detailed Description	268
5.128.2 Function/Subroutine Documentation	268
5.128.2.1 pesty()	268
5.129 pgen.f90 File Reference	268
5.129.1 Detailed Description	269
5.129.2 Function/Subroutine Documentation	269
5.129.2.1 pgen()	269
5.130 pgenhr.f90 File Reference	269
5.130.1 Detailed Description	269
5.131 pipeflow.f90 File Reference	270
5.131.1 Detailed Description	270
5.131.2 Function/Subroutine Documentation	270
5.131.2.1 pipeflow()	270
5.132 pkq.f90 File Reference	270
5.132.1 Detailed Description	271
5.132.2 Function/Subroutine Documentation	271
5.132.2.1 pkq()	271
5.133 plantmod.f90 File Reference	271
5.133.1 Detailed Description	271
5.133.2 Function/Subroutine Documentation	272
5.133.2.1 plantmod()	272
5.134 plantop.f90 File Reference	272
5.134.1 Detailed Description	272
5.134.2 Function/Subroutine Documentation	272
5.134.2.1 plantop()	272
5.135 pmeas.f90 File Reference	273
5.135.1 Detailed Description	273
5.135.2 Function/Subroutine Documentation	273
5.135.2.1 pmeas()	273
5.136 pminrl.f90 File Reference	273
5.136.1 Detailed Description	274
5.136.2 Function/Subroutine Documentation	274
5.136.2.1 pminrl()	274
5.137 pminrl2.f90 File Reference	274
5.137.1 Detailed Description	274

5.137.2 Function/Subroutine Documentation	274
5.137.2.1 pminrl2()	274
5.138 pond.f90 File Reference	275
5.138.1 Detailed Description	275
5.138.2 Function/Subroutine Documentation	275
5.138.2.1 pond()	275
5.139 pondhr.f90 File Reference	275
5.139.1 Detailed Description	276
5.139.2 Function/Subroutine Documentation	276
5.139.2.1 pondhr()	276
5.140 pothole.f90 File Reference	276
5.140.1 Detailed Description	276
5.140.2 Function/Subroutine Documentation	276
5.140.2.1 pothole()	277
5.141 print_hyd.f90 File Reference	278
5.141.1 Detailed Description	278
5.141.2 Function/Subroutine Documentation	278
5.141.2.1 print_hyd()	278
5.142 psed.f90 File Reference	278
5.142.1 Detailed Description	279
5.142.2 Function/Subroutine Documentation	279
5.142.2.1 psed()	279
5.143 qman.f90 File Reference	279
5.143.1 Detailed Description	279
5.143.2 Function/Subroutine Documentation	279
5.143.2.1 qman()	280
5.144 rchaa.f90 File Reference	280
5.144.1 Detailed Description	280
5.144.2 Function/Subroutine Documentation	280
5.144.2.1 rchaa()	280
5.145 rchday.f90 File Reference	281
5.145.1 Detailed Description	281
5.146 rchinit.f90 File Reference	281
5.146.1 Detailed Description	281
5.146.2 Function/Subroutine Documentation	281
5.146.2.1 rchinit()	281
5.147 rchmon.f90 File Reference	282
5.147.1 Detailed Description	282
5.147.2 Function/Subroutine Documentation	282
5.147.2.1 rchmon()	282
5.148 rchuse.f90 File Reference	282
5.148.1 Detailed Description	283

5.148.2 Function/Subroutine Documentation	283
5.148.2.1 <code>rchuse()</code>	283
5.149 <code>rchyr.f90</code> File Reference	283
5.149.1 Detailed Description	283
5.149.2 Function/Subroutine Documentation	283
5.149.2.1 <code>rchyr()</code>	283
5.150 <code>readatmodep.f90</code> File Reference	284
5.150.1 Detailed Description	284
5.151 <code>readbsn.f90</code> File Reference	284
5.151.1 Detailed Description	284
5.152 <code>readchm.f90</code> File Reference	284
5.152.1 Detailed Description	285
5.152.2 Function/Subroutine Documentation	285
5.152.2.1 <code>readchm()</code>	285
5.153 <code>readcnst.f90</code> File Reference	285
5.153.1 Detailed Description	285
5.153.2 Function/Subroutine Documentation	285
5.153.2.1 <code>readcnst()</code>	285
5.154 <code>readfcst.f90</code> File Reference	286
5.154.1 Detailed Description	286
5.155 <code>readfert.f90</code> File Reference	286
5.155.1 Detailed Description	286
5.156 <code>readfig.f90</code> File Reference	286
5.156.1 Detailed Description	287
5.157 <code>readfile.f90</code> File Reference	287
5.157.1 Detailed Description	287
5.158 <code>readgw.f90</code> File Reference	287
5.158.1 Detailed Description	287
5.158.2 Function/Subroutine Documentation	287
5.158.2.1 <code>readgw()</code>	287
5.159 <code>readhru.f90</code> File Reference	288
5.159.1 Detailed Description	288
5.159.2 Function/Subroutine Documentation	288
5.159.2.1 <code>readhru()</code>	288
5.160 <code>readinpt.f90</code> File Reference	288
5.160.1 Detailed Description	289
5.161 <code>readlup.f90</code> File Reference	289
5.161.1 Detailed Description	289
5.162 <code>readlwq.f90</code> File Reference	289
5.162.1 Detailed Description	289
5.162.2 Function/Subroutine Documentation	289
5.162.2.1 <code>readlwq()</code>	289



5.163 readmgt.f90 File Reference . . . . .	290
5.163.1 Detailed Description . . . . .	290
5.163.2 Function/Subroutine Documentation . . . . .	290
5.163.2.1 readmgt() . . . . .	290
5.164 readmon.f90 File Reference . . . . .	290
5.164.1 Detailed Description . . . . .	291
5.165 readops.f90 File Reference . . . . .	291
5.165.1 Detailed Description . . . . .	291
5.165.2 Function/Subroutine Documentation . . . . .	291
5.165.2.1 readops() . . . . .	291
5.166 readpest.f90 File Reference . . . . .	291
5.166.1 Detailed Description . . . . .	292
5.167 readplant.f90 File Reference . . . . .	292
5.167.1 Detailed Description . . . . .	292
5.168 readpnd.f90 File Reference . . . . .	292
5.168.1 Detailed Description . . . . .	292
5.168.2 Function/Subroutine Documentation . . . . .	292
5.168.2.1 readpnd() . . . . .	292
5.169 readres.f90 File Reference . . . . .	293
5.169.1 Detailed Description . . . . .	293
5.169.2 Function/Subroutine Documentation . . . . .	293
5.169.2.1 readres() . . . . .	293
5.170 readrte.f90 File Reference . . . . .	293
5.170.1 Detailed Description . . . . .	294
5.171 readru.f90 File Reference . . . . .	294
5.171.1 Detailed Description . . . . .	294
5.171.2 Function/Subroutine Documentation . . . . .	294
5.171.2.1 readru() . . . . .	294
5.172 readsdr.f90 File Reference . . . . .	294
5.172.1 Detailed Description . . . . .	295
5.172.2 Function/Subroutine Documentation . . . . .	295
5.172.2.1 readsdr() . . . . .	295
5.173 readsepticbz.f90 File Reference . . . . .	295
5.173.1 Detailed Description . . . . .	295
5.173.2 Function/Subroutine Documentation . . . . .	295
5.173.2.1 readsepticbz() . . . . .	296
5.174 readseptwq.f90 File Reference . . . . .	297
5.174.1 Detailed Description . . . . .	297
5.174.2 Function/Subroutine Documentation . . . . .	297
5.174.2.1 readseptwq() . . . . .	297
5.175 readsno.f90 File Reference . . . . .	297
5.175.1 Detailed Description . . . . .	298

5.175.2 Function/Subroutine Documentation	298
5.175.2.1 readsno()	298
5.176 readsol.f90 File Reference	298
5.176.1 Detailed Description	298
5.176.2 Function/Subroutine Documentation	298
5.176.2.1 readsol()	298
5.177 readsub.f90 File Reference	299
5.177.1 Detailed Description	299
5.177.2 Function/Subroutine Documentation	299
5.177.2.1 readsub()	299
5.178 readswq.f90 File Reference	299
5.178.1 Detailed Description	300
5.179 readtill.f90 File Reference	300
5.179.1 Detailed Description	300
5.180 readurban.f90 File Reference	300
5.180.1 Detailed Description	300
5.181 readwgn.f90 File Reference	300
5.181.1 Detailed Description	301
5.181.2 Function/Subroutine Documentation	301
5.181.2.1 readwgn()	301
5.182 readwus.f90 File Reference	301
5.182.1 Detailed Description	301
5.182.2 Function/Subroutine Documentation	301
5.182.2.1 readwus()	302
5.183 readwwq.f90 File Reference	302
5.183.1 Detailed Description	302
5.184 readyr.f90 File Reference	302
5.184.1 Detailed Description	302
5.184.2 Function/Subroutine Documentation	302
5.184.2.1 readyr()	302
5.185 reccnst.f90 File Reference	303
5.185.1 Detailed Description	303
5.185.2 Function/Subroutine Documentation	303
5.185.2.1 reccnst()	303
5.186 recday.f90 File Reference	303
5.186.1 Detailed Description	304
5.186.2 Function/Subroutine Documentation	304
5.186.2.1 recday()	304
5.187 rechour.f90 File Reference	304
5.187.1 Detailed Description	304
5.187.2 Function/Subroutine Documentation	304
5.187.2.1 rechour()	305

5.188 recmon.f90 File Reference . . . . .	306
5.188.1 Detailed Description . . . . .	306
5.188.2 Function/Subroutine Documentation . . . . .	306
5.188.2.1 recmon() . . . . .	306
5.189 recyear.f90 File Reference . . . . .	306
5.189.1 Detailed Description . . . . .	307
5.189.2 Function/Subroutine Documentation . . . . .	307
5.189.2.1 recyear() . . . . .	307
5.190 regres.f90 File Reference . . . . .	307
5.190.1 Detailed Description . . . . .	307
5.190.2 Function/Subroutine Documentation . . . . .	307
5.190.2.1 regres() . . . . .	307
5.191 res.f90 File Reference . . . . .	308
5.191.1 Detailed Description . . . . .	308
5.191.2 Function/Subroutine Documentation . . . . .	308
5.191.2.1 res() . . . . .	308
5.192 resetlu.f90 File Reference . . . . .	309
5.192.1 Detailed Description . . . . .	309
5.193 reshr.f90 File Reference . . . . .	309
5.193.1 Detailed Description . . . . .	309
5.193.2 Function/Subroutine Documentation . . . . .	309
5.193.2.1 reshr() . . . . .	309
5.194 resinit.f90 File Reference . . . . .	310
5.194.1 Detailed Description . . . . .	310
5.194.2 Function/Subroutine Documentation . . . . .	310
5.194.2.1 resinit() . . . . .	310
5.195 resnut.f90 File Reference . . . . .	310
5.195.1 Detailed Description . . . . .	310
5.195.2 Function/Subroutine Documentation . . . . .	311
5.195.2.1 resnut() . . . . .	311
5.196 rewind_init.f90 File Reference . . . . .	311
5.196.1 Detailed Description . . . . .	311
5.197 rhgen.f90 File Reference . . . . .	311
5.197.1 Detailed Description . . . . .	311
5.198 rootfr.f90 File Reference . . . . .	312
5.198.1 Detailed Description . . . . .	312
5.198.2 Function/Subroutine Documentation . . . . .	312
5.198.2.1 rootfr() . . . . .	312
5.199 route.f90 File Reference . . . . .	312
5.199.1 Detailed Description . . . . .	312
5.199.2 Function/Subroutine Documentation . . . . .	313
5.199.2.1 route() . . . . .	313

5.200 routels.f90 File Reference . . . . .	313
5.200.1 Detailed Description . . . . .	313
5.201 routeunit.f90 File Reference . . . . .	313
5.201.1 Detailed Description . . . . .	313
5.201.2 Function/Subroutine Documentation . . . . .	314
5.201.2.1 routeunit() . . . . .	314
5.202 routes.f90 File Reference . . . . .	314
5.202.1 Detailed Description . . . . .	314
5.202.2 Function/Subroutine Documentation . . . . .	314
5.202.2.1 routes() . . . . .	314
5.203 rsedaa.f90 File Reference . . . . .	315
5.203.1 Detailed Description . . . . .	315
5.203.2 Function/Subroutine Documentation . . . . .	315
5.203.2.1 rsedaa() . . . . .	315
5.204 rseday.f90 File Reference . . . . .	315
5.204.1 Detailed Description . . . . .	315
5.205 rsedmon.f90 File Reference . . . . .	316
5.205.1 Detailed Description . . . . .	316
5.205.2 Function/Subroutine Documentation . . . . .	316
5.205.2.1 rsedmon() . . . . .	316
5.206 rsedyr.f90 File Reference . . . . .	316
5.206.1 Detailed Description . . . . .	316
5.206.2 Function/Subroutine Documentation . . . . .	317
5.206.2.1 rsedyr() . . . . .	317
5.207 rtbact.f90 File Reference . . . . .	317
5.207.1 Detailed Description . . . . .	317
5.207.2 Function/Subroutine Documentation . . . . .	317
5.207.2.1 rtbact() . . . . .	317
5.208 rtday.f90 File Reference . . . . .	318
5.208.1 Detailed Description . . . . .	318
5.208.2 Function/Subroutine Documentation . . . . .	318
5.208.2.1 rtday() . . . . .	318
5.209 rteinit.f90 File Reference . . . . .	318
5.209.1 Detailed Description . . . . .	318
5.210 rthmusk.f90 File Reference . . . . .	319
5.210.1 Detailed Description . . . . .	319
5.210.2 Function/Subroutine Documentation . . . . .	319
5.210.2.1 rthmusk() . . . . .	319
5.211 rthpest.f90 File Reference . . . . .	319
5.211.1 Detailed Description . . . . .	319
5.211.2 Function/Subroutine Documentation . . . . .	320
5.211.2.1 rthpest() . . . . .	320

---

5.212 rthsed.f90 File Reference . . . . .	320
5.212.1 Detailed Description . . . . .	320
5.212.2 Function/Subroutine Documentation . . . . .	320
5.212.2.1 rthsed() . . . . .	320
5.213 rthvsc.f90 File Reference . . . . .	321
5.213.1 Detailed Description . . . . .	321
5.213.2 Function/Subroutine Documentation . . . . .	321
5.213.2.1 rthvsc() . . . . .	321
5.214 rtmusk.f90 File Reference . . . . .	321
5.214.1 Detailed Description . . . . .	322
5.214.2 Function/Subroutine Documentation . . . . .	322
5.214.2.1 rtmusk() . . . . .	322
5.215 rtout.f90 File Reference . . . . .	322
5.215.1 Detailed Description . . . . .	322
5.215.2 Function/Subroutine Documentation . . . . .	322
5.215.2.1 rtout() . . . . .	322
5.216 rtpest.f90 File Reference . . . . .	323
5.216.1 Detailed Description . . . . .	323
5.216.2 Function/Subroutine Documentation . . . . .	323
5.216.2.1 rtpest() . . . . .	323
5.217 rtsed.f90 File Reference . . . . .	323
5.217.1 Detailed Description . . . . .	324
5.217.2 Function/Subroutine Documentation . . . . .	324
5.217.2.1 rtsed() . . . . .	324
5.218 rtsed2.f90 File Reference . . . . .	324
5.218.1 Detailed Description . . . . .	324
5.218.2 Function/Subroutine Documentation . . . . .	324
5.218.2.1 rtsed2() . . . . .	325
5.219 sat_excess.f90 File Reference . . . . .	325
5.219.1 Detailed Description . . . . .	325
5.219.2 Function/Subroutine Documentation . . . . .	325
5.219.2.1 sat_excess() . . . . .	325
5.220 save.f90 File Reference . . . . .	326
5.220.1 Detailed Description . . . . .	326
5.220.2 Function/Subroutine Documentation . . . . .	326
5.220.2.1 save() . . . . .	326
5.221 saveconc.f90 File Reference . . . . .	326
5.221.1 Detailed Description . . . . .	326
5.221.2 Function/Subroutine Documentation . . . . .	327
5.221.2.1 saveconc() . . . . .	327
5.222 sched_mgt.f90 File Reference . . . . .	327
5.222.1 Detailed Description . . . . .	327

5.222.2 Function/Subroutine Documentation	327
5.222.2.1 sched_mgt()	327
5.223 schedule_ops.f90 File Reference	328
5.223.1 Detailed Description	328
5.223.2 Function/Subroutine Documentation	328
5.223.2.1 schedule_ops()	328
5.224 sim_initday.f90 File Reference	328
5.224.1 Detailed Description	329
5.225 sim_inityr.f90 File Reference	329
5.225.1 Detailed Description	329
5.226 simulate.f90 File Reference	329
5.226.1 Detailed Description	329
5.227 slrgen.f90 File Reference	329
5.227.1 Detailed Description	330
5.227.2 Function/Subroutine Documentation	330
5.227.2.1 slrgen()	330
5.228 smeas.f90 File Reference	330
5.228.1 Detailed Description	330
5.229 snom.f90 File Reference	330
5.229.1 Detailed Description	331
5.229.2 Function/Subroutine Documentation	331
5.229.2.1 snom()	331
5.230 soil_chem.f90 File Reference	331
5.230.1 Detailed Description	331
5.230.2 Function/Subroutine Documentation	331
5.230.2.1 soil_chem()	331
5.231 soil_phys.f90 File Reference	332
5.231.1 Detailed Description	332
5.231.2 Function/Subroutine Documentation	332
5.231.2.1 soil_phys()	332
5.232 soil_write.f90 File Reference	332
5.232.1 Detailed Description	333
5.232.2 Function/Subroutine Documentation	333
5.232.2.1 soil_write()	333
5.233 solp.f90 File Reference	333
5.233.1 Detailed Description	333
5.233.2 Function/Subroutine Documentation	333
5.233.2.1 solp()	333
5.234 solt.f90 File Reference	334
5.234.1 Detailed Description	334
5.235 std1.f90 File Reference	334
5.235.1 Detailed Description	334

5.236 std2.f90 File Reference . . . . .	334
5.236.1 Detailed Description . . . . .	335
5.237 std3.f90 File Reference . . . . .	335
5.237.1 Detailed Description . . . . .	335
5.238 stdaa.f90 File Reference . . . . .	335
5.238.1 Detailed Description . . . . .	335
5.239 storeinitial.f90 File Reference . . . . .	335
5.239.1 Detailed Description . . . . .	336
5.240 structure.f90 File Reference . . . . .	336
5.240.1 Detailed Description . . . . .	336
5.240.2 Function/Subroutine Documentation . . . . .	336
5.240.2.1 structure() . . . . .	336
5.241 sub_subbasin.f90 File Reference . . . . .	336
5.241.1 Detailed Description . . . . .	337
5.241.2 Function/Subroutine Documentation . . . . .	337
5.241.2.1 sub_subbasin() . . . . .	337
5.242 subaa.f90 File Reference . . . . .	337
5.242.1 Detailed Description . . . . .	337
5.242.2 Function/Subroutine Documentation . . . . .	337
5.242.2.1 subaa() . . . . .	337
5.243 subbasin.f90 File Reference . . . . .	338
5.243.1 Detailed Description . . . . .	338
5.243.2 Function/Subroutine Documentation . . . . .	338
5.243.2.1 subbasin() . . . . .	338
5.244 subday.f90 File Reference . . . . .	338
5.244.1 Detailed Description . . . . .	339
5.244.2 Function/Subroutine Documentation . . . . .	339
5.244.2.1 subday() . . . . .	339
5.245 submon.f90 File Reference . . . . .	339
5.245.1 Detailed Description . . . . .	339
5.246 substor.f90 File Reference . . . . .	339
5.246.1 Detailed Description . . . . .	340
5.246.2 Function/Subroutine Documentation . . . . .	340
5.246.2.1 substor() . . . . .	340
5.247 subwq.f90 File Reference . . . . .	340
5.247.1 Detailed Description . . . . .	340
5.247.2 Function/Subroutine Documentation . . . . .	340
5.247.2.1 subwq() . . . . .	340
5.248 subyr.f90 File Reference . . . . .	341
5.248.1 Detailed Description . . . . .	341
5.249 sumhyd.f90 File Reference . . . . .	341
5.249.1 Detailed Description . . . . .	341

5.250 sumv.f90 File Reference . . . . .	341
5.250.1 Detailed Description . . . . .	342
5.250.2 Function/Subroutine Documentation . . . . .	342
5.250.2.1 sumv() . . . . .	342
5.251 surface.f90 File Reference . . . . .	342
5.251.1 Detailed Description . . . . .	342
5.251.2 Function/Subroutine Documentation . . . . .	342
5.251.2.1 surface() . . . . .	342
5.252 surfst_h2o.f90 File Reference . . . . .	343
5.252.1 Detailed Description . . . . .	343
5.252.2 Function/Subroutine Documentation . . . . .	343
5.252.2.1 surfst_h2o() . . . . .	343
5.253 surfstor.f90 File Reference . . . . .	343
5.253.1 Detailed Description . . . . .	344
5.253.2 Function/Subroutine Documentation . . . . .	344
5.253.2.1 surfstor() . . . . .	344
5.254 surq_daycn.f90 File Reference . . . . .	344
5.254.1 Detailed Description . . . . .	344
5.254.2 Function/Subroutine Documentation . . . . .	344
5.254.2.1 surq_daycn() . . . . .	345
5.255 surq_greenampt.f90 File Reference . . . . .	345
5.255.1 Detailed Description . . . . .	345
5.255.2 Function/Subroutine Documentation . . . . .	345
5.255.2.1 surq_greenampt() . . . . .	345
5.256 swbl.f90 File Reference . . . . .	346
5.256.1 Detailed Description . . . . .	346
5.256.2 Function/Subroutine Documentation . . . . .	346
5.256.2.1 swbl() . . . . .	346
5.257 sweep.f90 File Reference . . . . .	346
5.257.1 Detailed Description . . . . .	346
5.257.2 Function/Subroutine Documentation . . . . .	347
5.257.2.1 sweep() . . . . .	347
5.258 swu.f90 File Reference . . . . .	347
5.258.1 Detailed Description . . . . .	347
5.258.2 Function/Subroutine Documentation . . . . .	347
5.258.2.1 swu() . . . . .	347
5.259 tair.f90 File Reference . . . . .	348
5.259.1 Detailed Description . . . . .	348
5.259.2 Function/Subroutine Documentation . . . . .	348
5.259.2.1 tair() . . . . .	348
5.260 tgen.f90 File Reference . . . . .	349
5.260.1 Detailed Description . . . . .	349



5.260.2 Function/Subroutine Documentation	349
5.260.2.1 tgen()	349
5.261 theta.f90 File Reference	349
5.261.1 Detailed Description	350
5.261.2 Function/Subroutine Documentation	350
5.261.2.1 theta()	350
5.262 tillfactor.f90 File Reference	350
5.262.1 Detailed Description	350
5.262.2 Function/Subroutine Documentation	351
5.262.2.1 tillfactor()	351
5.263 tmeas.f90 File Reference	351
5.263.1 Detailed Description	351
5.264 tran.f90 File Reference	352
5.264.1 Detailed Description	352
5.264.2 Function/Subroutine Documentation	352
5.264.2.1 tran()	352
5.265 transfer.f90 File Reference	352
5.265.1 Detailed Description	352
5.265.2 Function/Subroutine Documentation	353
5.265.2.1 transfer()	353
5.266 tstr.f90 File Reference	353
5.266.1 Detailed Description	353
5.266.2 Function/Subroutine Documentation	353
5.266.2.1 tstr()	353
5.267 ttcoef.f90 File Reference	354
5.267.1 Detailed Description	354
5.267.2 Function/Subroutine Documentation	354
5.267.2.1 ttcoef()	354
5.268 ttcoef_wway.f90 File Reference	354
5.268.1 Detailed Description	354
5.269 urb_bmp.f90 File Reference	355
5.269.1 Detailed Description	355
5.269.2 Function/Subroutine Documentation	355
5.269.2.1 urb_bmp()	355
5.270 urban.f90 File Reference	355
5.270.1 Detailed Description	355
5.270.2 Function/Subroutine Documentation	356
5.270.2.1 urban()	356
5.271 urbanhr.f90 File Reference	356
5.271.1 Detailed Description	356
5.271.2 Function/Subroutine Documentation	356
5.271.2.1 urbanhr()	356

5.272 varinit.f90 File Reference . . . . .	357
5.272.1 Detailed Description . . . . .	357
5.272.2 Function/Subroutine Documentation . . . . .	357
5.272.2.1 varinit() . . . . .	357
5.273 vbl.f90 File Reference . . . . .	357
5.273.1 Detailed Description . . . . .	358
5.273.2 Function/Subroutine Documentation . . . . .	358
5.273.2.1 vbl() . . . . .	358
5.274 virtual.f90 File Reference . . . . .	359
5.274.1 Detailed Description . . . . .	359
5.274.2 Function/Subroutine Documentation . . . . .	359
5.274.2.1 virtual() . . . . .	359
5.275 volq.f90 File Reference . . . . .	359
5.275.1 Detailed Description . . . . .	360
5.275.2 Function/Subroutine Documentation . . . . .	360
5.275.2.1 volq() . . . . .	360
5.276 washp.f90 File Reference . . . . .	360
5.276.1 Detailed Description . . . . .	360
5.276.2 Function/Subroutine Documentation . . . . .	360
5.276.2.1 washp() . . . . .	360
5.277 watbal.f90 File Reference . . . . .	361
5.277.1 Detailed Description . . . . .	361
5.277.2 Function/Subroutine Documentation . . . . .	361
5.277.2.1 watbal() . . . . .	361
5.278 water_hru.f90 File Reference . . . . .	361
5.278.1 Detailed Description . . . . .	362
5.278.2 Function/Subroutine Documentation . . . . .	362
5.278.2.1 water_hru() . . . . .	362
5.279 watqual.f90 File Reference . . . . .	362
5.279.1 Detailed Description . . . . .	362
5.279.2 Function/Subroutine Documentation . . . . .	362
5.279.2.1 watqual() . . . . .	362
5.280 watqual2.f90 File Reference . . . . .	363
5.280.1 Detailed Description . . . . .	363
5.280.2 Function/Subroutine Documentation . . . . .	363
5.280.2.1 watqual2() . . . . .	363
5.281 wattable.f90 File Reference . . . . .	363
5.281.1 Detailed Description . . . . .	364
5.282 watuse.f90 File Reference . . . . .	364
5.282.1 Detailed Description . . . . .	364
5.282.2 Function/Subroutine Documentation . . . . .	364
5.282.2.1 watuse() . . . . .	364

5.283 weatgn.f90 File Reference . . . . .	365
5.283.1 Detailed Description . . . . .	365
5.283.2 Function/Subroutine Documentation . . . . .	365
5.283.2.1 weatgn() . . . . .	365
5.284 wetlan.f90 File Reference . . . . .	365
5.284.1 Detailed Description . . . . .	365
5.284.2 Function/Subroutine Documentation . . . . .	366
5.284.2.1 wetlan() . . . . .	366
5.285 wmeas.f90 File Reference . . . . .	366
5.285.1 Detailed Description . . . . .	366
5.286 wndgen.f90 File Reference . . . . .	366
5.286.1 Detailed Description . . . . .	366
5.286.2 Function/Subroutine Documentation . . . . .	367
5.286.2.1 wndgen() . . . . .	367
5.287 writea.f90 File Reference . . . . .	367
5.287.1 Detailed Description . . . . .	367
5.287.2 Function/Subroutine Documentation . . . . .	367
5.287.2.1 writea() . . . . .	367
5.288 writeaa.f90 File Reference . . . . .	368
5.288.1 Detailed Description . . . . .	368
5.289 writed.f90 File Reference . . . . .	368
5.289.1 Detailed Description . . . . .	368
5.290 writem.f90 File Reference . . . . .	368
5.290.1 Detailed Description . . . . .	368
5.290.2 Function/Subroutine Documentation . . . . .	368
5.290.2.1 writem() . . . . .	368
5.291 xmon.f90 File Reference . . . . .	369
5.291.1 Detailed Description . . . . .	369
5.292 ysed.f90 File Reference . . . . .	369
5.292.1 Detailed Description . . . . .	369
5.292.2 Function/Subroutine Documentation . . . . .	369
5.292.2.1 ysed() . . . . .	369
5.293 zero0.f90 File Reference . . . . .	370
5.293.1 Detailed Description . . . . .	370
5.294 zero1.f90 File Reference . . . . .	370
5.294.1 Detailed Description . . . . .	370
5.295 zero2.f90 File Reference . . . . .	370
5.295.1 Detailed Description . . . . .	371
5.296 zero_urban.f90 File Reference . . . . .	371
5.296.1 Detailed Description . . . . .	371
5.297 zeroini.f90 File Reference . . . . .	371
5.297.1 Detailed Description . . . . .	371

<b>Bibliography</b>	<b>373</b>
<b>Index</b>	<b>375</b>

# Chapter 1

## SWAT

An upgraded SWAT 2012 revision 670 code

### Objectives

- Standard indentation and translation to Fortran 90 by using `findent`. See the `translate-fortran90.pl` perl script file (:heavy\_check\_mark:)
- Exhaustive use of the "implicit none" directive to detect bad variable usage (:heavy\_check\_mark:)
- Generate a GNU `Make` makefile and compile with GNU `GFortran`. See the `generate-makefile.pl` perl script file (:heavy\_check\_mark:)
- Remove non-used local variables and format labels (:heavy\_check\_mark:)
- Remove non-used global variables (:construction:)
- Detect and solve all uninitialized variables (:heavy\_check\_mark: :construction:, some proposed solutions could be incorrect)
- Remove unneeded variable initializations (:heavy\_check\_mark:) as:  

```
j=0 ! this line is not necessary  
j=ihru
```
- Remove redundant code (:heavy\_check\_mark:)
- Exhaustive use of the "parameter" directive on constants (:heavy\_check\_mark:)
- Remove global counters (as `i`, `ihru`, `iihru`, `inum1` or `idum` in module `parm`). Using local counters or passing values as argument are preferred (:construction:)
- Generate a detailed list of issues detected in the original code (:heavy\_check\_mark:, see at the end of this README)
- Remove obsolete commented code (:x:)
- Update variable descriptions in comments (:construction:, a lot of work)
- Standardize comments by using Doxygen style in order to generate documentation. See at `latex/refman.pdf` (:heavy\_check\_mark:)

## Required tools

- `GFortran` (to compile the source code)
- `Make` (to build the executable file)
- `Perl` (optional: to execute the perl scripts to update the makefile or to translate original files to Fortran 90)
- `Findent` (optional: to translate original files to Fortran 90 with a standard indentation)
- `Doxygen` (optional: to generate a reference programming manual from source code)
- `TeX Live` or `MiKTeX` (optional: to generate a reference programming manual from source code)
- On Microsoft Windows systems you have to install `MSYS2` and the required utilities ( `GFortran` and `Make`). You can follow detailed instructions in `install-unix`

## Instructions to generate Fortran 90 style code from original code

In order to generate Fortran 90 style code with standard indentation from original code you have to type on a UNIX type terminal (you need `Perl` and `Findent`):

```
$ perl translate-fortran90.pl
```

## Instructions to generate an initial GNU make Makefile

Type on the UNIX type terminal, when translated the original code to Fortran 90 style (you need `Perl`):

```
$ perl generate-makefile.pl
```

## Instructions to generate an executable to test

Type on the UNIX type terminal (you need `GFortran` and `Make`)

- In UNIX type operative systems:

```
$ make
```

- In a `MSYS2` terminal in Microsoft Windows:

```
$ EXE=".exe" LDFLAGS="-static" make
```

- Cross-compiling a 32 bits Microsoft Windows executable in a UNIX type operative system:

```
$ prefix="i686-w64-mingw32-" EXE=".exe" LDFLAGS="-static" make
```

- Cross-compiling a 64 bits Microsoft Windows executable in a UNIX type operative system:

```
$ prefix="x86_64-w64-mingw32-" EXE=".exe" LDFLAGS="-static" make
```

## Instructions to generate an optimized executable file

Type on the UNIX type terminal (you need **GFortran** and **Make**)

- In UNIX type operative systems:

```
$ CFLAGS="-march=native -flto" LDFLAGS="-flto" make strip
```

- In a **MSYS2** terminal in Microsoft Windows:

```
$ EXE=".exe" CFLAGS="-flto" LDFLAGS="-flto -static" make strip
```

- Cross-compiling a 32 bits Microsoft Windows executable in a UNIX type operative system:

```
$ prefix="i686-w64-mingw32-" EXE=".exe" CFLAGS="-flto" LDFLAGS="-flto -static" make strip
```

- Cross-compiling a 64 bits Microsoft Windows executable in a UNIX type operative system:

```
$ prefix="x86_64-w64-mingw32-" EXE=".exe" CFLAGS="-flto" LDFLAGS="-flto -static" make strip
```

## Instructions to generate a reference programming manual from source code

Type on the UNIX type terminal (you need **Doxygen** and **TeX Live** or **MiKTeX**):

```
$ make latex/refman.pdf
```

The reference programming manual file latex/refman.pdf is generated from source code in PDF format

## Issues in the original source code

This is a list of possible issues detected in the original source code. These issues have been mostly detected by the **GFortran** compiler warnings. Some of them could not arise because the logic of the variables is not possible.

- In biofilm.f:

- dcoef is used but not initialized. dcoef=3 as in watqual.f? Then, I propose at beginning: `real*8, parameter :: dcoef = 3.`

- In bmp\_ri\_pond.f:

- qseep and qet could be used not initialized at lines 133 and 134. However the problem only arises for `nstep<1`

- In bmp\_sand\_filter.f:

- sed\_removed at line 342 could be used not initialized if `sfsedstdev<=0`

- In bpm\_sed\_pond.f:

- bmp\_sed\_pond seems to be bmp\_sed\_pond at line 186

- In bmp\_wet\_pond.f:

- hvol could be used not initialized in ext\_dpth subroutine at line 267 in first bucle iteration

- In `clicon.f`:
  - `tmxbsb`, `tmnbsb`, `rbsb`, `rstpbsb`, `rhdbbsb`, `rabsb`, `rmxbsb`, `daylbsb`, `fradbsb` and `u10bsb` could be used not initialized at 186-207 lines
- In `conapply.f`:
  - `k` and `kk` could be used not initialized at 121-122 lines if `iday_pest(j) /= ipst_freq(j)` and `curyr > nyskip`
- In `confert.f`:
  - `ifrt` seems to be it at line 214
- In `curno.f`:
  - `smxold` could be used not initialized if `cn1(h) <= 1.e-6` and `curyr /= 0` at line 96
- In `depstor.f`
  - `itill` is used at line 64 but not initialized in any part of code
- In `drains.f`:
  - `nlayer` could be used not initialized at line 23. However, the problem only arises if it is not set in the previous bucle (`mlyr <= 1` or `sol_z(j1, j) <= 0`)
- In `etact.f`:
  - `sev` could be used not initialized at line 286 if `dep >= esd` and `ly == 2`
- In `filter.f`:
  - `remove21` seems to be `remove2` at line 316
- In `grass_wway.f`:
  - `sf_depth` and `sf_sed` could be used not initialized at lines 133 and 137 if `sf_area > 0` and `sf_area <= 1.e-6`
- In `headout.f`:
  - `hedr` array of column titles is written out of defined bounds at lines 118, 119, 121 and 133. It is written to `mrcho` (set to 62 in `allocate_parms.f` line 59) but in `modparm.f` the bound of `hedr` array is set to 46 (line 663)
- In `hhnoqual.f`:
  - `algon` seems to be `algcon` at line 190
- In `hhwatqual.f`
  - `orgnpin` seems to be `orgpin` at line 278
  - `thour = 1.0` at line 377 overwrites previous `thour` calculation. It is wrong
- In `hmeas.f`:
  - `rhdbbsb` could be used not initialized at line 84
- In `hruaa.f`:
  - `pdvas(70) = wtabelo` at line 249 but `wtabelo` is not initialized in any part of code
- In `killop.f`:
  - `ff1` and `ff2` are used but not initialized at lines 167 and 267. They are set in `harvkillop.f` file (lines 257-258). They have to be included in `modparm.f` to share `harvkillop.f` values? or they have to be redefined as in `harvkillop.f`?



- In NCsed\_leach.f90:
  - `perc_clyr` could be used not initialized at line 221 if `sol_nly(j) < 2`
- In nrain.f:
  - `no2pcp` seems to be `no3pcp` at line 72
- In pmeas.f:
  - `rbsb` could be used not initialized at line 143
  - `flag` could be used not initialized if `'a=='` at line 210 –`rainsb` could be used not initialized, however only if `nstep <= 0`
- In pminrl2.f:
  - at line 95 a comma is necessary between `base` and `vara`
  - `ssp` could be used not initialized at line 196 if `xx <= 1.e-6`
- In pothole.f:
  - `solp_tileo` could be used not initialized at line 593 if `pot_vol(j) <= 1.e-6` or `potvol_↔ tile <= 1.e-6`
- In potholehr.f:
  - `potflow` seems to be `potflwo` at line 447
- In readatmodep.f:
  - `momax=12*nbyr` is defined at line 65 but not used. It has to be `mo_max`? but then, it overwrites the file read
- In readops.f:
  - `year = 0.` seems to be `iyear = 0` at line 98
  - `mg13` seems to be `mgt13` at line 206
- In readpnd.f:
  - `vselsetlpnd` seems to be `velsetlpnd` at line 279
- In readru.f:
  - `tck` is used but not initialized at line 79
- In readsepticbz.f:
  - at line 135 `4.e-8` seems to be `4.e-8`
- In resbact.f:
  - `reswtr` is used at lines 78, 79 and 89 but it is not initialized in any part of code
- In rewind\_init.f:
  - `orig_tnylda` is used but not initialized at line 174
- In routels.f:
  - `dstor` is used but not initialized at line 134. It has to be calculated as in `watbal.f`? or as in the commented line 109?
  - `latqout` and `gwqout` could be used not initialized at lines 142-143
- In rtbact.f:
  - `netwtr` could be used not initialized at line 124, however only if `nstep < 1`

- In `rthpest.f`:
  - `thour=1.0` at line 183 overwrites previous `thour` calculation. It is wrong
  - `frsol` and `frsrb` could be used not initialized at lines 289-290 if `hrtwtr(ii)>0.001` and `hrtwtr(ii)/(idt*60)<=0.01`
- In `rtpest.f`:
  - `tday=1.0` at line 180 overwrites previous `tday` calculation. It is wrong
- In `sched_mgt.f`:
  - `< =` seems to be `<=` at 202 line
  - `husc` and `igrow` at lines 264-265 are used but not initialized. `husc` has to be `phu_op(iop, ihru)` has in `readmgt.f?` `igrow` has to be `igro(ihru)` has in `readmgt.f?`
- In `schedule_ops.f`:
  - `so_res_flag(iops, ihru)` is used but not initialized at line 149
  - `so_res(iops, ihru)` is used but not initialized at line 150
- In `smeas.f`:
  - `rabsb` could be used not initialized at line 86
- In `sweep.f`:
  - `fr_curb` is used but not initialized at line 56. It has to be added to `modparm.f` to share result with `sched_mgt.f?` or it has to be `mgt5op(nop(ihru), ihru)` as in `sched_mgt.f?`
- In `tillfactor.f`
  - `tillagef(1, jj)` could be used but not initialized at line 44
- In `tmeas.f`:
  - `tmxbsb` and `tmnbsb` could be used not initialized at lines 109-110
- In `transfer.f`:
  - `ratio, xx` and `ratio1` could be used not initialized at lines 236, 239 and 241 if `ihout==2`
- In `urbanhr.f`
  - `isweep(j)` is used at lines 166 and 186 but not initialized at any part of code
- In `watqual2.f`
  - `wattemp` is used but not initialized at line 271
- In `wmeas.f`:
  - `u10bsb` could be used not initialized at line 85
- In `zero0.f`:
  - `sol_sumn03` seems to be `sol_sumno3` at line 508
- In `zero_urban.f`:
  - `stp_stagdis` seems to be `dtp_stagdis` at line 84
  - `subdr_kg` seems to be `subdr_km` at line 149
  - `spl_eros` is not defined at line 21, it could be `eros_spl?`

## Chapter 2

# Modules Index

### 2.1 Modules List

Here is a list of all documented modules with brief descriptions:

<a href="#">parm</a>	Main module containing the global variables . . . . .	<a href="#">15</a>
----------------------	---	--------------------



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">addh.f90</a>	101
<a href="#">albedo.f90</a>	102
<a href="#">allocate_parms.f90</a>	102
<a href="#">alph.f90</a>	103
<a href="#">anfert.f90</a>	103
<a href="#">apex_day.f90</a>	104
<a href="#">apply.f90</a>	105
<a href="#">ascrv.f90</a>	106
<a href="#">atri.f90</a>	107
<a href="#">aunif.f90</a>	107
<a href="#">autoirr.f90</a>	108
<a href="#">bacteria.f90</a>	109
<a href="#">biozone.f90</a>	110
<a href="#">bmp_det_pond.f90</a>	110
<a href="#">bmp_ri_pond.f90</a>	111
<a href="#">bmp_sand_filter.f90</a>	111
<a href="#">bmp_sed_pond.f90</a>	112
<a href="#">bmp_wet_pond.f90</a>	112
<a href="#">bmpinit.f90</a>	113
<a href="#">buffer.f90</a>	114
<a href="#">burnop.f90</a>	114
<a href="#">canopyint.f90</a>	115
<a href="#">caps.f90</a>	116
<a href="#">carbon_new.f90</a>	116
<a href="#">carbon_zhang2.f90</a>	117
<a href="#">cfactor.f90</a>	118
<a href="#">clgen.f90</a>	119
<a href="#">clicon.f90</a>	119
<a href="#">command.f90</a>	120
<a href="#">conapply.f90</a>	121
<a href="#">confert.f90</a>	122
<a href="#">crackflow.f90</a>	122
<a href="#">crackvol.f90</a>	123
<a href="#">curno.f90</a>	124
<a href="#">dailycn.f90</a>	124

decay.f90	125
depstor.f90	126
distributed_bmps.f90	126
dormant.f90	127
drains.f90	127
dstn1.f90	128
ee.f90	130
eiusle.f90	131
enrsb.f90	131
estimate_ksat.f90	132
etact.f90	133
etpot.f90	133
expo.f90	134
fcgd.f90	134
fert.f90	135
filter.f90	135
filtw.f90	136
finalbal.f90	137
gcycl.f90	137
getallo.f90	137
grass_wway.f90	138
graze.f90	138
grow.f90	139
gw_no3.f90	140
gwmod.f90	140
gwmod_deep.f90	141
gwnutr.f90	142
h2omgt_init.f90	142
harvestop.f90	143
harvkillop.f90	143
headout.f90	144
hhnoqual.f90	144
hhwatqual.f90	145
hmeas.f90	146
HQDAV.f90	146
hruaa.f90	146
hruallo.f90	147
hruday.f90	147
hrumon.f90	148
hrupond.f90	148
hrupondhr.f90	149
hruyr.f90	150
hydroinit.f90	150
icl.f90	150
impnd_init.f90	151
impndaa.f90	151
impndday.f90	152
impndmon.f90	153
impndyr.f90	153
irr_rch.f90	153
irr_res.f90	154
irrigate.f90	155
irrsb.f90	155
jdt.f90	156
killop.f90	158
lakeq.f90	158
latsed.f90	159
layersplit.f90	160

lid_cistern.f90	160
lid_greenroof.f90	161
lid_por pavement.f90	162
lid_raingarden.f90	162
lidinit.f90	163
lids.f90	164
log_normal.f90	165
lwqdef.f90	166
main.f90	166
modparm.f90	167
ndenit.f90	250
newtillmix.f90	251
nfix.f90	252
nitvol.f90	253
nlch.f90	253
nminrl.f90	254
noqual.f90	255
npup.f90	255
nrain.f90	256
nup.f90	257
nuts.f90	257
openwth.f90	258
operatn.f90	259
orgn.f90	259
orgncswat.f90	260
orgncswat2.f90	261
origtile.f90	262
ovr_sed.f90	262
oxygen_saturation.f90	263
percmacro.f90	264
percmmain.f90	264
percmicro.f90	265
pestlch.f90	267
pestw.f90	267
pesty.f90	268
pgen.f90	268
pgenhr.f90	269
pipeflow.f90	270
pkq.f90	270
plantmod.f90	271
plantop.f90	272
pmeas.f90	273
pminrl.f90	273
pminrl2.f90	274
pond.f90	275
pondhr.f90	275
pothole.f90	276
print_hyd.f90	278
psed.f90	278
qman.f90	279
rchaa.f90	280
rchday.f90	281
rchinit.f90	281
rchmon.f90	282
rchuse.f90	282
rchyr.f90	283
readatmodep.f90	284
readbsn.f90	284

<a href="#">readchm.f90</a>	<a href="#">284</a>
<a href="#">readcnst.f90</a>	<a href="#">285</a>
<a href="#">readfcst.f90</a>	<a href="#">286</a>
<a href="#">readfert.f90</a>	<a href="#">286</a>
<a href="#">readfig.f90</a>	<a href="#">286</a>
<a href="#">readfile.f90</a>	<a href="#">287</a>
<a href="#">readgw.f90</a>	<a href="#">287</a>
<a href="#">readhru.f90</a>	<a href="#">288</a>
<a href="#">readingpt.f90</a>	<a href="#">288</a>
<a href="#">readlup.f90</a>	<a href="#">289</a>
<a href="#">readlwq.f90</a>	<a href="#">289</a>
<a href="#">readmgt.f90</a>	<a href="#">290</a>
<a href="#">readmon.f90</a>	<a href="#">290</a>
<a href="#">readops.f90</a>	<a href="#">291</a>
<a href="#">readpest.f90</a>	<a href="#">291</a>
<a href="#">readplant.f90</a>	<a href="#">292</a>
<a href="#">readpnd.f90</a>	<a href="#">292</a>
<a href="#">readres.f90</a>	<a href="#">293</a>
<a href="#">readrte.f90</a>	<a href="#">293</a>
<a href="#">readru.f90</a>	<a href="#">294</a>
<a href="#">readsdr.f90</a>	<a href="#">294</a>
<a href="#">readsepticbz.f90</a>	<a href="#">295</a>
<a href="#">readseptwq.f90</a>	<a href="#">297</a>
<a href="#">readsno.f90</a>	<a href="#">297</a>
<a href="#">readsol.f90</a>	<a href="#">298</a>
<a href="#">readsub.f90</a>	<a href="#">299</a>
<a href="#">readswq.f90</a>	<a href="#">299</a>
<a href="#">readtill.f90</a>	<a href="#">300</a>
<a href="#">readurban.f90</a>	<a href="#">300</a>
<a href="#">readwgn.f90</a>	<a href="#">300</a>
<a href="#">readwus.f90</a>	<a href="#">301</a>
<a href="#">readwwq.f90</a>	<a href="#">302</a>
<a href="#">readyr.f90</a>	<a href="#">302</a>
<a href="#">reccnst.f90</a>	<a href="#">303</a>
<a href="#">recday.f90</a>	<a href="#">303</a>
<a href="#">rechour.f90</a>	<a href="#">304</a>
<a href="#">recmon.f90</a>	<a href="#">306</a>
<a href="#">recyear.f90</a>	<a href="#">306</a>
<a href="#">regres.f90</a>	<a href="#">307</a>
<a href="#">res.f90</a>	<a href="#">308</a>
<a href="#">resetlu.f90</a>	<a href="#">309</a>
<a href="#">reshr.f90</a>	<a href="#">309</a>
<a href="#">resinit.f90</a>	<a href="#">310</a>
<a href="#">resnut.f90</a>	<a href="#">310</a>
<a href="#">rewind_init.f90</a>	<a href="#">311</a>
<a href="#">rhgen.f90</a>	<a href="#">311</a>
<a href="#">rootfr.f90</a>	<a href="#">312</a>
<a href="#">route.f90</a>	<a href="#">312</a>
<a href="#">routels.f90</a>	<a href="#">313</a>
<a href="#">routeunit.f90</a>	<a href="#">313</a>
<a href="#">routres.f90</a>	<a href="#">314</a>
<a href="#">rsedaa.f90</a>	<a href="#">315</a>
<a href="#">rseday.f90</a>	<a href="#">315</a>
<a href="#">rsedmon.f90</a>	<a href="#">316</a>
<a href="#">rsedyr.f90</a>	<a href="#">316</a>
<a href="#">rtbact.f90</a>	<a href="#">317</a>
<a href="#">rtday.f90</a>	<a href="#">318</a>
<a href="#">rteinit.f90</a>	<a href="#">318</a>



rthmusk.f90	319
rthpest.f90	319
rthsed.f90	320
rthvsc.f90	321
rtmusk.f90	321
rtout.f90	322
rtpest.f90	323
rtsed.f90	323
rtsed2.f90	324
sat_excess.f90	325
save.f90	326
saveconc.f90	326
sched_mgt.f90	327
schedule_ops.f90	328
sim_initday.f90	328
sim_inityr.f90	329
simulate.f90	329
slrgen.f90	329
smeas.f90	330
snom.f90	330
soil_chem.f90	331
soil_phys.f90	332
soil_write.f90	332
solp.f90	333
solt.f90	334
std1.f90	334
std2.f90	334
std3.f90	335
stdaa.f90	335
storeinitial.f90	335
structure.f90	336
sub_subbasin.f90	336
subaa.f90	337
subbasin.f90	338
subday.f90	338
submon.f90	339
substor.f90	339
subwq.f90	340
subyr.f90	341
sumhyd.f90	341
sumv.f90	341
surface.f90	342
surfst_h2o.f90	343
surfstor.f90	343
surq_daycn.f90	344
surq_greenampt.f90	345
swbl.f90	346
sweep.f90	346
swu.f90	347
tair.f90	348
tgen.f90	349
theta.f90	349
tillfactor.f90	350
tmeas.f90	351
tran.f90	352
transfer.f90	352
tstr.f90	353
ttcoef.f90	354

ttcoef_wway.f90	354
urb_bmp.f90	355
urban.f90	355
urbanhr.f90	356
varinit.f90	357
vbl.f90	357
virtual.f90	359
volq.f90	359
washp.f90	360
watbal.f90	361
water_hru.f90	361
watqual.f90	362
watqual2.f90	363
watable.f90	363
watuse.f90	364
weatgn.f90	365
wetlan.f90	365
wmeas.f90	366
wndgen.f90	366
writea.f90	367
writeaa.f90	368
writed.f90	368
writem.f90	368
xmon.f90	369
ysed.f90	369
zero0.f90	370
zero1.f90	370
zero2.f90	370
zero_urban.f90	371
zeroini.f90	371

## Chapter 4

# Module Documentation

### 4.1 parm Module Reference

main module containing the global variables

#### Variables

- integer, parameter **mvaro** = 33  
*max number of variables routed through the reach*
- integer, parameter **mhruo** = 79  
*maximum number of variables written to HRU output file (output.hru) (none)*
- integer, parameter **mrcho** = 62  
*maximum number of variables written to reach output file (.rch) (none)*
- integer, parameter **msubo** = 24  
*maximum number of variables written to subbasin output file (output.sub) (none)*
- integer, parameter **mstdo** = 113  
*max number of variables summarized in output.std*
- integer, parameter **motot** = 600
- character(len=80), parameter **prog** = "SWAT Sep 7 VER 2018/Rev 670"  
*SWAT program header string (name and version)*
- character(len=13), dimension(**mhruo**), parameter **heds** = (/ "PRECIPmm", "SNOFALLmm", "SNOMELTmm", "IRRmm", "PETmm", "ETmm", "SW\_INITmm", "SW\_ENDmm", "PERCmm", "GW\_RCHGmm", "DA\_RCHGmm", "REVAPmm", "SA\_IRRmm", "DA\_IRRmm", "SA\_STmm", "DA\_STmm", "SURQ\_GENmm", "SURQ\_CNTmm", "TLOSSmm", "LATQGENmm", "GW\_Qmm", "WYLDmm", "DAILYCN", "TMP\_AVdgC", "TMP\_MXdgC", "TMP\_MNdgc", "SOL\_TMPdgC", "SOLARMJ/m2", "SYLDt/ha", "USLEt/ha", "N\_APPkg/ha", "P\_APPkg/ha", "NAUTOkg/ha", "PAUTOkg/ha", "NGRZkg/ha", "PGRZkg/ha", "NCFRTkg/ha", "PCFRTkg/ha", "NRAINkg/ha", "NFIKkg/ha", "F-MNkg/ha", "A-MNkg/ha", "A-SNkg/ha", "F-MPkg/ha", "AO-LPkg/ha", "L-APkg/ha", "A-SPkg/ha", "DNITkg/ha", "NUPkg/ha", "PUPkg/ha", "ORGNkg/ha", "ORGPkg/ha", "SEDPkg/ha", "NSURQkg/ha", "NLATQkg/ha", "NO3Lkg/ha", "NO3GWkg/ha", "SOLPkg/ha", "P\_GWkg/ha", "W\_STRS", "TMP\_STRS", "N\_STRS", "P\_STRS", "BIOMt/ha", "LAI", "YLDt/ha", "BACTPct", "BACTLPct", "WTAB CLIm", "WTAB SOLm", "SNOmm", "CMUPkg/ha", "CMTOTkg/ha", "QTILEmm", "TNO3kg/ha", "LNO3kg/ha", "GW\_QDmm", "LATQCNTmm", "TVAPkg/ha"/)  
*column headers for HRU output file*
- character(len=13), dimension(**msubo**), parameter **hedb** = (/ "PRECIPmm", "SNOMELTmm", "PETmm", "ETmm", "SWmm", "PERCmm", "SURQmm", "GW\_Qmm", "WYLDmm", "SYLDt/ha", "ORGNkg/ha", "ORGPkg/ha", "NSURQkg/ha", "SOLPkg/ha", "SEDPkg/ha", "LAT Q(mm)", "LATNO3kg/h", "GWNO3kg/ha", "CHOLAmic/L", "CBODU mg/L", "DOXQ mg/L", "TNO3kg/ha", "QTILEmm", "TVAPkg/ha"/)



- integer **icalen**  
code for writing out calendar day or julian day to output.rch, .sub, .hru files;  
icalen = 0 (print julian day), 1 (print month/day/year);  
icalen MUST be == zero if IPRINT == 3 to print subdaily
- real \*8 **prf\_bsn**  
Basinwide peak rate adjustment factor for sediment routing in the channel. Allows impact of peak flow rate on sediment routing and channel reshaping to be taken into account.
- real \*8 **co2\_x2**
- real \*8 **co2\_x**
- real \*8, dimension(:), allocatable **cdn**  
denitrification exponential rate coefficient
- real \*8, dimension(:), allocatable **nperco**  
nitrate percolation coefficient (0-1)  
0:concentration of nitrate in surface runoff is zero  
1:percolate has same concentration of nitrate as surface runoff
- real \*8, dimension(:), allocatable **surlag**  
Surface runoff lag time. This parameter is needed in subbasins where the time of concentration is greater than 1 day. SURLAG is used to create a "storage" for surface runoff to allow the runoff to take longer than 1 day to reach the subbasin outlet (days)
- real \*8, dimension(:), allocatable **cmn**  
rate factor for humus mineralization on active organic N
- real \*8, dimension(:), allocatable **phoskd**  
phosphorus soil partitioning coefficient. Ratio of soluble phosphorus in surface layer attached to sediment to phosphorus dissolved in soil water
- real \*8, dimension(:), allocatable **psp**  
phosphorus availability index. The fraction of fertilizer P remaining in labile pool after initial rapid phase of P sorption (none)
- real \*8, dimension(:), allocatable **sdnco**  
denitrification threshold: fraction of field capacity triggering denitrification
- real \*8 **pst\_kg**  
amount of pesticide applied to HRU (kg/ha)
- real \*8 **yield**  
yield (dry weight) (kg)
- real \*8 **burn\_frlb**  
fraction of biomass and residue that burn(input in management file) range (0 - 1.0) (none)
- real \*8 **yieldgrn**
- real \*8 **yieldbms**
- real \*8 **yieldtbr**
- real \*8 **yieldn**
- real \*8 **yieldp**
- real \*8 **hi\_bms**
- real \*8 **hi\_rsd**
- real \*8 **yieldrsd**
- real \*8, dimension(:,:), allocatable **hru\_rufr**
- real \*8, dimension(:,:), allocatable **daru\_km**
- real \*8, dimension(:,:), allocatable **ru\_k**
- real \*8, dimension(:,:), allocatable **ru\_c**
- real \*8, dimension(:,:), allocatable **ru\_eiq**
- real \*8, dimension(:,:), allocatable **ru\_ovsl**
- real \*8, dimension(:,:), allocatable **ru\_a**
- real \*8, dimension(:,:), allocatable **ru\_ovs**
- real \*8, dimension(:,:), allocatable **ru\_ktc**
- real \*8, dimension(:), allocatable **gwq\_ru**
- real \*8, dimension(:), allocatable **qdayout**

- integer, dimension(:), allocatable **ils2**
- integer, dimension(:), allocatable **ils2flag**
- integer **ipest**  
*pesticide identification number from pest.dat (none)*
- integer **iru**
- integer **mru**
- integer **irch**
- integer **isub**
- integer **mhyd\_bsn**
- integer **ils\_nofig**
- integer **mhru1**
- real \*8 **wshd\_sepno3**
- real \*8 **wshd\_sepnh3**
- real \*8 **wshd\_seporgn**
- real \*8 **wshd\_sepfon**
- real \*8 **wshd\_seporgp**
- real \*8 **wshd\_sepfop**
- real \*8 **wshd\_sepsolp**
- real \*8 **wshd\_sepbod**
- real \*8 **wshd\_sepmm**
- integer, dimension(:), allocatable **isep\_hru**
- real \*8 **fixco**  
*nitrogen fixation coefficient*
- real \*8 **nfixmx**  
*maximum daily n-fixation (kg/ha)*
- real \*8 **res\_stlr\_co**  
*reservoir sediment settling coefficient*
- real \*8 **rsd\_covco**  
*residue cover factor for computing fraction of cover*
- real \*8 **vcrit**  
*critical velocity*
- real \*8 **wshd\_snob**  
*average amount of water stored in snow at the beginning of the simulation for the entire watershed (mm H2O)*
- real \*8 **wshd\_sw**  
*water in soil at beginning of simulation, or\ average amount of water stored in soil for the entire watershed, or\ difference between mass balance calculated from watershed averages and actual value for water in soil at end of simulation (goal is to have wshd\_sw = 0.) (mm H2O)*
- real \*8 **wshd\_pndfr**  
*fraction of watershed area which drains into ponds (none)*
- real \*8 **wshd\_pndsed**  
*total amount of suspended sediment in ponds in the watershed (metric tons), or mass balance discrepancy for pond sediment expressed as loading per unit hectare of drainage area (metric tons/ha)*
- real \*8 **wshd\_pndv**  
*total volume of water in ponds in the watershed ( $m^3$ ), or mass balance discrepancy for pond water volume expressed as depth over drainage area (mm H2O)*
- real \*8 **percop**  
*pesticide percolation coefficient (0-1)*  
*0: concentration of pesticide in surface runoff is zero*  
*1: percolate has same concentration of pesticide as surface runoff*
- real \*8 **wshd\_resfr**  
*fraction of watershed area that drains into reservoirs (none)*
- real \*8 **wshd\_pndha**

- watershed area in hectares which drains into ponds (ha)*

  - real \*8 **wshd\_resha**
- watershed area in hectares which drains into reservoirs (ha)*

  - real \*8 **wshd\_fminp**
- average annual amount of mineral P applied in watershed (kg P/ha)*

  - real \*8 **wshd\_fnh3**
- average annual amount of NH3-N applied in watershed (kg N/ha)*

  - real \*8 **wshd\_fno3**
- average annual amount of NO3-N applied in watershed (kg N/ha)*

  - real \*8 **wshd\_forgn**
- average annual amount of organic N applied in watershed (kg N/ha)*

  - real \*8 **wshd\_ftotn**
- average annual amount of N (mineral & organic) applied in watershed (kg N/ha)*

  - real \*8 **wshd\_forgp**
- average annual amount of organic P applied in watershed (kg P/ha)*

  - real \*8 **wshd\_ftotp**
- average annual amount of P (mineral & organic) applied in watershed (kg P/ha)*

  - real \*8 **wshd\_yldn**
- amount of nitrogen removed from soil in watershed in the yield (kg N/ha)*

  - real \*8 **wshd\_yldp**
- amount of phosphorus removed from soil in watershed in the yield (kg P/ha)*

  - real \*8 **wshd\_fixn**
- average annual amount of nitrogen added to plant biomass via fixation (kg N/ha)*

  - real \*8 **wshd\_pup**
- average annual amount of plant uptake of phosphorus (kg P/ha)*

  - real \*8 **wshd\_nstrs**
- average annual number of nitrogen stress units in watershed (stress units)*

  - real \*8 **wshd\_pstrs**
- average annual number of phosphorus stress units in watershed (stress units)*

  - real \*8 **wshd\_tstrs**
- average annual number of temperature stress units in watershed (stress units)*

  - real \*8 **wshd\_wstrs**
- average annual number of water stress units in watershed (stress units)*

  - real \*8 **wshd\_astrs**
- initial soil water content expressed as a fraction of field capacity*

  - real \*8 **ffcb**
- average annual amount of nitrogen lost from nitrate pool due to denitrification in watershed (kg N/ha)*

  - real \*8 **wshd\_dnit**
- average annual amount of nitrogen moving from active organic to nitrate pool in watershed (kg N/ha)*

  - real \*8 **wshd\_hmn**
- average annual amount of phosphorus moving from organic to labile pool in watershed (kg P/ha)*

  - real \*8 **wshd\_hmp**
- average annual amount of nitrogen moving from fresh organic (residue) to nitrate and active organic pools in watershed (kg N/ha)*

  - real \*8 **wshd\_rmn**
- average annual amount of nitrogen moving from active organic to stable organic pool in watershed (kg N/ha)*

  - real \*8 **wshd\_rwn**
- die-off factor for persistent bacteria in soil solution (1/day)*

  - real \*8 **wdpq**
- average annual amount of nitrogen moving from active organic to stable organic pool in watershed (kg N/ha)*

  - real \*8 **wshd\_rmp**

- average annual amount of phosphorus moving from fresh organic (residue) to labile and organic pools in watershed (kg P/ha)*

  - real \*8 **wshd\_nitn**
- average annual amount of nitrogen moving from the NH<sub>3</sub> to the NO<sub>3</sub> pool by nitrification in the watershed (kg N/ha)*

  - real \*8 **wshd\_voln**
- average annual amount of nitrogen lost by ammonia volatilization in watershed (kg N/ha)*

  - real \*8 **wshd\_pal**
- average annual amount of phosphorus moving from labile mineral to active mineral pool in watershed (kg P/ha)*

  - real \*8 **wshd\_pas**
- average annual amount of phosphorus moving from active mineral to stable mineral pool in watershed (kg P/ha)*

  - real \*8 **wof\_p**
- fraction of persistent bacteria on foliage that is washed off by a rainfall event (none)*

  - real \*8 **wshd\_raino3**
- average annual amount of NO<sub>3</sub> added to soil by rainfall in watershed (kg N/ha)*

  - real \*8 **wshd\_plch**
- average annual amount of phosphorus leached into second soil layer (kg P/ha)*

  - real \*8 **ressedc**
- net change in sediment in reservoir during day (metric tons)*

  - real \*8 **basno3f**
- final average amount of nitrogen in the nitrate pool in watershed soil (kg N/ha)*

  - real \*8 **basorgnf**
- final average amount of nitrogen in the organic N pool in watershed soil (kg N/ha)*

  - real \*8 **wshd\_pinlet**
- real \*8 **wshd\_ptile***
- real \*8 **sftmp***

*Snowfall temperature (deg C)*
- real \*8 **smfmn***

*Minimum melt rate for snow during year (Dec. 21) where deg C refers to the air temperature. (mm/deg C/day)*
- real \*8 **smfmx***

*Maximum melt rate for snow during year (June 21) where deg C refers to the air temperature. SMFMX and SMFMN allow the rate of snow melt to vary through the year. These parameters are accounting for the impact of soil temperature on snow melt. (mm/deg C/day)*
- real \*8 **smtmp***

*Snow melt base temperature. Mean air temperature at which snow melt will occur. (deg C)*
- real \*8 **basminpf***

*final average amount of phosphorus in the mineral P pool in watershed soil (kg P/ha)*
- real \*8 **basorgpf***

*final average amount of phosphorus in the organic P pool in watershed soil (kg P/ha)*
- real \*8 **wshd\_ressed***

*total amount of suspended sediment in reservoirs in the watershed (metric tons), or mass balance discrepancy for reservoir sediment expressed as loading per unit hectare of drainage area (metric tons/ha)*
- real \*8 **wshd\_resv***

*total volume of water in all reservoirs in the watershed (m<sup>3</sup>), or mass balance discrepancy for reservoir water volume expressed as depth over drainage area (mm H<sub>2</sub>O)*
- real \*8 **basminpi***

*average amount of phosphorus initially in the mineral P pool in watershed soil (kg P/ha)*
- real \*8 **basno3i***

*average amount of nitrogen initially in the nitrate pool in watershed soil (kg N/ha)*
- real \*8 **basorgni***

*average amount of nitrogen initially in the organic N pool in watershed soil (kg N/ha)*
- real \*8 **basorgpi***



- average amount of phosphorus initially in the organic P pool in watershed soil (kg P/ha)*

  - real \*8 **peakr**

*peak runoff rate for the day in HRU or channel ( $m^3/s$ )*
- real \*8 **albd**

*albedo of ground for the day in HRU, the fraction of the solar radiation reflected at the soil surface back into space (none)*
- real \*8 **pndsedin**

*sediment inflow to the pond from HRU during day (metric tons)*
- real \*8 **sw\_excess**

*amount of water stored in soil layer on the current day that exceeds field capacity (gravity drained water) (mm H<sub>2</sub>O)*
- real \*8 **timp**

*Snow pack temperature lag factor (0-1)*  
*1 = no lag (snow pack temp=current day air temp) as the lag factor goes to zero, the snow pack's temperature will be less influenced by the current day's air temperature.*
- real \*8 **wt\_shall**

*shallow water table depth above the impervious layer (mm H<sub>2</sub>O)*
- real \*8 **sq\_rto**
- real \*8 **qtile**

*amount of water in drainage tile flow in HRU soil layer for the day (mm H<sub>2</sub>O)*
- real \*8 **infilpcp**

*amount of precipitation that infiltrates into soil (enters soil) (mm H<sub>2</sub>O)*
- real \*8 **fixn**

*amount of nitrogen added to the plant biomass via fixation on the day in HRU (kg N/ha)*
- real \*8 **latlyr**

*amount of water in lateral flow in layer in HRU for the day (mm H<sub>2</sub>O)*
- real \*8 **snofall**

*amount of precipitation falling as freezing rain/snow on day in HRU (mm H<sub>2</sub>O)*
- real \*8 **snomlt**

*amount of water in snow melt for the day in HRU (mm H<sub>2</sub>O)*
- real \*8 **tloss**

*amount of water removed from surface runoff via transmission losses on day in HRU (mm H<sub>2</sub>O)*
- real \*8 **lpndloss**
- real \*8 **lwetloss**
- real \*8 **bioday**

*biomass generated on current day in HRU (kg)*
- real \*8 **cfertn**

*total amount of nitrogen applied to soil during continuous fertilizer operation in HRU on day (kg N/ha)*
- real \*8 **cfertp**

*amount of phosphorus applied to soil during continuous fertilizer operation in HRU on day (kg P/ha)*
- real \*8 **fertn**

*total amount of nitrogen applied to soil in HRU on day in fertilizer application (kg N/ha)*
- real \*8 **sepd**

*micropore percolation from bottom of the soil layer on day in HRU (mm H<sub>2</sub>O)*
- real \*8 **sol\_rd**

*current rooting depth (mm)*
- real \*8 **sedrch**

*sediment transported out of channel or reach during time step (metric tons)*
- real \*8 **sepcrktot**
- real \*8 **fertno3**
- real \*8 **fertnh3**
- real \*8 **fertorgn**
- real \*8 **fertsolp**

- real \*8 **fertorgp**
- real \*8 **qdftr**  
*fraction of water yield that is surface runoff (none)*
- real \*8 **ferftp**  
*total amount of phosphorus applied to soil in HRU on day in fertilizer application (kg P/ha)*
- real \*8 **grazn**  
*amount of nitrogen added to soil in grazing on the day in HRU (kg N/ha)*
- real \*8 **grazp**  
*amount of phosphorus added to soil in grazing on the day in HRU (kg P/ha)*
- real \*8 **soxy**  
*saturation dissolved oxygen concentration (mg/L)*
- real \*8 **rtwtr**  
*water leaving reach on day ( $m^3 H_2O$ )*
- real \*8 **sdti**  
*average flow rate in reach for day ( $m^3/s$ )*
- real \*8 **ressa**  
*surface area of reservoir on day (ha)*
- real \*8 **da\_km**  
*area of the watershed in square kilometers ( $km^2$ )*
- real \*8 **rchdep**  
*depth of flow on day (m)*
- real \*8 **rtevp**  
*evaporation from reach on day ( $m^3 H_2O$ )*
- real \*8 **rttime**  
*reach travel time (hour)*
- real \*8 **rttlc**  
*transmission losses from reach on day ( $m^3 H_2O$ )*
- real \*8 **resflwi**  
*water entering reservoir on day ( $m^3 H_2O$ )*
- real \*8 **wdprch**  
*die-off factor for persistent bacteria in streams (1/day)*
- real \*8 **resev**  
*evaporation from reservoir on day ( $m^3 H_2O$ )*
- real \*8 **resflwo**  
*water leaving reservoir on day ( $m^3 H_2O$ )*
- real \*8 **respcp**  
*precipitation on reservoir for day ( $m^3 H_2O$ )*
- real \*8 **ressedi**  
*sediment entering reservoir during time step (metric tons)*
- real \*8 **ressedo**  
*sediment leaving reservoir during time step (metric tons)*
- real \*8 **ressep**  
*seepage from reservoir on day ( $m^3 H_2O$ )*
- real \*8 **pperco\_bsn**  
*phosphorus percolation coefficient. Ratio of soluble phosphorus in surface to soluble phosphorus in percolate*
- real \*8 **nperco\_bsn**  
*basin nitrate percolation coefficient (0-1)*  
*0:concentration of nitrate in surface runoff is zero*  
*1:percolate has same concentration of nitrate as surface runoff*
- real \*8 **rsdco**

- residue decomposition coefficient. The fraction of residue which will decompose in a day assuming optimal moisture, temperature, C:N ratio, and C:P ratio*
- real \*8 **voltot**  
*total volume of cracks expressed as depth per unit area (mm)*
  - real \*8 **phoskd\_bsn**
  - real \*8 **msk\_x**  
*weighting factor controlling relative importance of inflow rate and outflow rate in determining storage on reach*
  - real \*8 **volcrmin**  
*minimum crack volume allowed in any soil layer (mm), or  
minimum soil volume in profile (mm)*
  - real \*8 **bactkdq**  
*bacteria soil partitioning coefficient. Ratio of solution bacteria in surface layer to solution bacteria in runoff soluble and sorbed phase in surface runoff.*
  - real \*8 **canev**  
*amount of water evaporated from canopy storage (mm H2O)*
  - real \*8 **precipday**  
*precipitation, or effective precipitation reaching soil surface, for the current day in HRU (mm H2O)*
  - real \*8 **uno3d**  
*plant nitrogen deficiency for day in HRU (kg N/ha)*
  - real \*8 **usle**  
*daily soil loss predicted with USLE equation (metric tons/ha)*
  - real \*8 **rcn**  
*concentration of nitrogen in the rainfall (mg/L)*
  - real \*8 **surlag\_bsn**
  - real \*8 **thbact**  
*temperature adjustment factor for bacteria die-off/growth*
  - real \*8 **wlpq20**  
*overall rate change for less persistent bacteria in soil solution (1/day)*
  - real \*8 **wlps20**  
*overall rate change for less persistent bacteria adsorbed to soil particles (1/day)*
  - real \*8 **wpq20**  
*overall rate change for persistent bacteria in soil solution (1/day)*
  - real \*8 **wps20**  
*overall rate change for persistent bacteria adsorbed to soil particles (1/day)*
  - real \*8 **bactrop**  
*persistent bacteria transported to main channel with surface runoff (# colonies/ha)*
  - real \*8 **bactsedp**  
*persistent bacteria transported with sediment in surface runoff (# colonies/ha)*
  - real \*8 **enratio**  
*enrichment ratio calculated for current day in HRU (none)*
  - real \*8 **pndpcp**  
*precipitation on pond during day ( $m^3$  H2O)*
  - real \*8 **wetpcp**  
*precipitation on wetland for day ( $m^3$  H2O)*
  - real \*8 **wetsep**  
*seepage from wetland bottom for day ( $m^3$  H2O)*
  - real \*8 **pndev**  
*evaporation from pond on day ( $m^3$  H2O)*
  - real \*8 **pndflwi**  
*volume of water flowing into pond on day ( $m^3$  H2O)*
  - real \*8 **pndsedo**

- sediment leaving pond during day (metric tons)*
- real \*8 **pndsep**
  - seepage from pond on day ( $m^3 H_2O$ )*
- real \*8 **wetev**
  - evaporation from wetland for day ( $m^3 H_2O$ )*
- real \*8 **wetflwi**
  - volume of water flowing in wetland on day ( $m^3 H_2O$ )*
- real \*8 **wetsedo**
  - sediment loading from wetland for day (metric tons)*
- real \*8 **da\_ha**
  - drainage area of watershed in hectares (ha)*
- real \*8 **pndflwo**
  - volume of water flowing out of pond on day ( $m^3 H_2O$ )*
- real \*8 **vpd**
  - vapor pressure deficit (kPa)*
- real \*8 **wetflwo**
  - volume of water flowing out wetland on day ( $m^3 H_2O$ )*
- real \*8 **wetsedi**
  - sediment loading to wetland for day (metric tons)*
- real \*8 **evlai**
  - leaf area index at which no evaporation occurs. This variable is used in ponded HRUs (eg rice) where evaporation from the water surface is restricted by the plant canopy cover. Evaporation from the water surface equals potential ET when LAI = 0 and decreased linearly to 0 when LAI = EVLAI*
- real \*8 **evrch**
  - Reach evaporation adjustment factor. Evaporation from the reach is multiplied by EVRCH. This variable was created to limit the evaporation predicted in arid regions.*
- real \*8 **ep\_day**
  - actual amount of transpiration that occurs on day in HRU (mm H<sub>2</sub>O)*
- real \*8 **pet\_day**
  - potential evapotranspiration on current day in HRU (mm H<sub>2</sub>O)*
- real \*8 **bactrolp**
  - less persistent bacteria transported to main channel with surface runoff (# colonies/ha)*
- real \*8 **bactsedlp**
  - less persistent bacteria transported with sediment in surface runoff (# colonies/ha)*
- real \*8 **adj\_pkr**
  - peak rate adjustment factor in the subbasin. Used in the MUSLE equation to account for impact of peak flow on erosion (none)*
- real \*8 **n\_updis**
  - nitrogen uptake distribution parameter. This parameter controls the amount of nitrogen removed from the different soil layer layers by the plant. In particular, this parameter allows the amount of nitrogen removed from the surface layer via plant uptake to be controlled. While the relationship between UBN and N removed from the surface layer is affected by the depth of the soil profile, in general, as UBN increases the amount of N removed from the surface layer relative to the amount removed from the entire profile increases*
- real \*8 **nactfr**
  - nitrogen active pool fraction. The fraction of organic nitrogen in the active pool (none)*
- real \*8 **p\_updis**
  - phosphorus uptake distribution parameter This parameter controls the amount of phosphorus removed from the different soil layers by the plant. In particular, this parameter allows the amount of phosphorus removed from the surface layer via plant uptake to be controlled. While the relationship between UBP and P uptake from the surface layer is affected by the depth of the soil profile, in general, as UBP increases the amount of P removed from the surface layer relative to the amount removed from the entire profile increases*
- real \*8 **snoev**
  - amount of water in snow lost through sublimation on current day in HRU (mm H<sub>2</sub>O)*

- real \*8 **sno3up**  
*amount of nitrate moving upward in the soil profile in watershed (kg N/ha)*
- real \*8 **reactw**  
*amount of pesticide in lake water of reach that is lost through reactions (mg pst)*
- real \*8 **es\_day**  
*actual amount of evaporation (soil et) that occurs on day in HRU (mm H2O)*
- real \*8 **sdiegro1pq**  
*average annual change in the number of less persistent bacteria colonies in soil solution in watershed (# cfu/m<sup>2</sup>)*
- real \*8 **sdiegro1ps**  
*average annual change in the number of less persistent bacteria colonies on soil particles in watershed (# cfu/m<sup>2</sup>)*
- real \*8 **sdiegro2pq**  
*average annual change in the number of persistent bacteria colonies in soil solution in watershed (# cfu/m<sup>2</sup>)*
- real \*8 **sdiegro2ps**  
*average annual change in the number of persistent bacteria colonies on soil particles in watershed (# cfu/m<sup>2</sup>)*
- real \*8 **wof\_lp**  
*fraction for less persistent bacteria on foliage that is washed off by a rainfall event (none)*
- real \*8 **ep\_max**  
*maximum amount of transpiration (plant et) that can occur on day in HRU (mm H2O)*
- real \*8 **sbactrolp**  
*average annual number of less persistent bacteria transported to main channel with surface runoff in solution (# colonies/ha)*
- real \*8 **sbactrop**  
*average annual number of persistent bacteria transported to main channel with surface runoff in solution (# colonies/ha)*
- real \*8 **sbactsedlp**  
*average annual number of less persistent bacteria transported with sediment in surface runoff (# colonies/ha)*
- real \*8 **sbactsedp**  
*average annual number of persistent bacteria transported with sediment in surface runoff (# colonies/ha)*
- real \*8 **sbactlchlp**  
*average annual number of less persistent bacteria lost from soil surface layer by percolation (# cfu/m<sup>2</sup>)*
- real \*8 **sbactlchp**  
*average annual number of persistent bacteria lost from soil surface layer by percolation (# cfu/m<sup>2</sup>)*
- real \*8 **rchwtr**  
*water stored in reach at beginning of day (m<sup>3</sup> H2O)*
- real \*8 **resuspst**  
*amount of pesticide moving from sediment to reach due to resuspension (mg pst)*
- real \*8 **setlpst**  
*amount of pesticide moving from water to sediment due to settling (mg pst)*
- real \*8 **psp\_bsn**
- real \*8 **bsprev**  
*surface runoff lagged from prior day of simulation (mm H2O)*
- real \*8 **bssprev**  
*lateral flow lagged from prior day of simulation (mm H2O)*
- real \*8 **spadyev**  
*average annual amount of water removed from potholes by evaporation in watershed (mm H2O)*
- real \*8 **spadyo**  
*average annual amount of water released to main channel from potholes in watershed (mm H2O)*
- real \*8 **spadyrfv**  
*average annual amount of precipitation on potholes in watershed (mm H2O)*
- real \*8 **spadyssp**  
*average annual amount of water removed from potholes by seepage in watershed (mm H2O)*

- real \*8 **spadyosp**
- real \*8 **qday**  
amount of surface runoff loading to main channel from HRU on current day (includes effects of transmission losses) (mm H2O)
- real \*8 **al5**  
fraction of total rainfall that occurs during 0.5h of highest intensity rain (none)
- real \*8 **no3pcp**  
nitrate added to the soil in rainfall (kg N/ha)
- real \*8 **pndsedc**  
net change in sediment in pond during day (metric tons)
- real \*8 **usle\_ei**  
USLE rainfall erosion index on day for HRU (100(ft·tn in)/(acre·hr))
- real \*8 **rcharea**  
cross-sectional area of flow (m<sup>2</sup>)
- real \*8 **volatpst**  
amount of pesticide lost from lake water of reach by volatilization (mg pst)
- real \*8 **ubw**  
water uptake distribution parameter. This parameter controls the amount of water removed from the different soil layers by the plant. In particular, this parameter allows the amount of water removed from the surface layer via plant uptake to be controlled. While the relationship between UBW and H2O removed from the surface layer is affected by the depth of the soil profile, in general, as UBW increases the amount of water removed from the surface layer relative to the amount removed from the entire profile increases
- real \*8 **uobn**  
nitrogen uptake normalization parameter. This variable normalizes the nitrogen uptake so that the model can easily verify that uptake from the different soil layers sums to 1.0
- real \*8 **uobp**  
phosphorus uptake normalization parameter. This variable normalizes the phosphorus uptake so that the model can easily verify that uptake from the different soil layers sums to 1.0
- real \*8 **uobw**  
water uptake normalization parameter. This variable normalizes the water uptake so that the model can easily verify that uptake from the different soil layers sums to 1.0
- real \*8 **wetsedc**  
net change in sediment in wetland during day (metric tons)
- real \*8 **respesti**  
pesticide entering reservoir on day (mg pst)
- real \*8 **rcor**  
correction coefficient for generated rainfall to ensure that the annual means for generated and observed values are comparable (needed only if IDIST=1)
- real \*8 **rexp**  
value of exponent for mixed exponential rainfall distribution (needed only if IDIST=1)
- real \*8 **snocov1**  
1st shape parameter for snow cover equation. This parameter is determined by solving the equation for 50% snow cover
- real \*8 **snocov2**  
2nd shape parameter for snow cover equation. This parameter is determined by solving the equation for 95% snow cover
- real \*8 **snocovmx**  
Minimum snow water content that corresponds to 100% snow cover. If the snow water content is less than SNOC<sub>←</sub> OVMX, then a certain percentage of the ground will be bare (mm H2O)
- real \*8 **ai0**  
ratio of chlorophyll-a to algal biomass (ug chl-a/mg alg)
- real \*8 **ai1**  
fraction of algal biomass that is nitrogen (mg N/mg alg)

- real \*8 [ai2](#)  
*fraction of algal biomass that is phosphorus (mg P/mg alg)*
- real \*8 [ai3](#)  
*the rate of oxygen production per unit of algal photosynthesis (mg O2/mg alg)*
- real \*8 [ai4](#)  
*the rate of oxygen uptake per unit of algae respiration (mg O2/mg alg)*
- real \*8 [ai5](#)  
*the rate of oxygen uptake per unit of NH3 nitrogen oxidation (mg O2/mg N)*
- real \*8 [ai6](#)  
*the rate of oxygen uptake per unit of NO2 nitrogen oxidation (mg O2/mg N)*
- real \*8 [rhoq](#)  
*algal respiration rate at 20 deg C (1/day or 1/hr)*
- real \*8 [tfact](#)  
*fraction of solar radiation computed in the temperature heat balance that is photosynthetically active*
- real \*8 [k\\_l](#)  
*half-saturation coefficient for light (MJ/(m2\*hr))*
- real \*8 [k\\_n](#)  
*michaelis-menton half-saturation constant for nitrogen (mg N/L)*
- real \*8 [k\\_p](#)  
*michaelis-menton half saturation constant for phosphorus (mg P/L)*
- real \*8 [lambda0](#)  
*non-algal portion of the light extinction coefficient (1/m)*
- real \*8 [lambda1](#)  
*linear algal self-shading coefficient (1/(m\*ug chla/L))*
- real \*8 [lambda2](#)  
*nonlinear algal self-shading coefficient ((1/m)(ug chla/L)\*\*(-2/3))*
- real \*8 [mumax](#)  
*maximum specific algal growth rate at 20 deg C(1/day or 1/hr)*
- real \*8 [p\\_n](#)  
*algal preference factor for ammonia*
- real \*8 [rnum1](#)  
*variable to hold value for rnum1s(:) (none)*
- real \*8 [etday](#)  
*actual evapotranspiration occurring on day in HRU (mm H2O)*
- real \*8 [auton](#)  
*amount of nitrogen applied in auto-fert application (kg N/ha)*
- real \*8 [autop](#)  
*amount of phosphorus applied in auto-fert application (kg P/ha)*
- real \*8 [hmntl](#)  
*amount of nitrogen moving from active organic to nitrate pool in soil profile on current day in HRU (kg N/ha)*
- real \*8 [hmptl](#)  
*amount of phosphorus moving from active organic to nitrate pool in soil profile on current day in HRU (kg P/ha)*
- real \*8 [rmn2tl](#)  
*amount of nitrogen moving from the fresh organic (residue) to the nitrate(80%) and active organic(20%) pools in soil profile on current day in HRU (kg N/ha)*
- real \*8 [rwntl](#)  
*amount of nitrogen moving from active organic to stable organic pool in soil profile on current day in HRU (kg N/ha)*
- real \*8 [gwseep](#)  
*amount of water recharging deep aquifer on current day in HRU (mm H2O)*
- real \*8 [revapday](#)

- amount of water moving from the shallow aquifer into the soil profile or being taken up by plant roots in the shallow aquifer or in the bank storage zone (mm H<sub>2</sub>O)*

  - real \*8 **rmp1tl**
- amount of phosphorus moving from the labile mineral pool to the active mineral pool in the soil profile on the current day in the HRU (kg P/ha)*

  - real \*8 **rmptl**
- amount of phosphorus moving from the fresh organic (residue) to the labile(80%) and organic(20%) pools in soil profile on current day in HRU (kg P/ha)*

  - real \*8 **roctl**
- amount of phosphorus moving from the active mineral pool to the stable mineral pool in the soil profile on the current day in the HRU (kg P/ha)*

  - real \*8 **wdntl**
- amount of nitrogen lost from nitrate pool by denitrification in soil profile on current day in HRU (kg N/ha)*

  - real \*8 **cmn\_bsn**
- die-off factor for less persistent bacteria in streams (1/day)*

  - real \*8 **wdlprch**
- die-off factor for persistent bacteria in reservoirs (1/day)*

  - real \*8 **wdpres**
- potential ET value read in for day (mm H<sub>2</sub>O)*

  - real \*8 **petmeas**
- loss of pesticide from active sediment layer by burial (mg pst)*

  - real \*8 **bury**
- diffusion of pesticide from sediment to reach lake water (mg pst)*

  - real \*8 **difus**
- amount of pesticide in sediment that is lost through reactions (mg pst)*

  - real \*8 **reactb**
- soluble pesticide concentration in outflow on day (mg pst/m<sup>3</sup>)*

  - real \*8 **solpesto**
- die-off factor for less persistent bacteria in reservoirs (1/day)*

  - real \*8 **wdlpres**
- sorbed pesticide concentration in outflow on day (mg pst/m<sup>3</sup>)*

  - real \*8 **sorpesto**
- soluble pesticide entering reservoir (mg pst)*

  - real \*8 **solpesti**
- sorbed pesticide entering reservoir (mg pst)*

  - real \*8 **sorpesti**
- calibration coefficient to control impact of the storage time constant for the reach at bankfull depth (phi(10,:)) upon the storage time constant for the reach used in the Muskingum flow method*

  - real \*8 **spcon\_bsn**
- calibration coefficient to control impact of the storage time constant for the reach at 0.1 bankfull depth (phi(13,:)) upon the storage time constant for the reach used in the Muskingum flow method*

  - real \*8 **spexp\_bsn**
- calibration coefficient to control impact of the storage time constant for the reach at bankfull depth (phi(10,:)) upon the storage time constant for the reach used in the Muskingum flow method*

  - real \*8 **msk\_co1**
- calibration coefficient to control impact of the storage time constant for the reach at 0.1 bankfull depth (phi(13,:)) upon the storage time constant for the reach used in the Muskingum flow method*

  - real \*8 **msk\_co2**
- depth of water in deep aquifer in HRU (mm H<sub>2</sub>O)*

  - real \*8 **deepstp**
- depth of water in shallow aquifer in HRU on previous day (mm H<sub>2</sub>O)*

  - real \*8 **shallstp**
- amount of water stored as snow on previous day (mm H<sub>2</sub>O)*

  - real \*8 **snoprev**
- amount of water stored in soil profile in the HRU on the previous day (mm H<sub>2</sub>O)*

  - real \*8 **swprev**



- real \*8 **reschlao**  
*amount of chlorophyll-a leaving reservoir on day (kg chl-a)*
- real \*8 **resno2o**  
*amount of nitrite leaving reservoir on day (kg N)*
- real \*8 **resno3o**  
*amount of nitrate leaving reservoir on day (kg N)*
- real \*8 **resorgno**  
*amount of organic N leaving reservoir on day (kg N)*
- real \*8 **resorgpo**  
*amount of organic P leaving reservoir on day (kg P)*
- real \*8 **ressolpo**  
*amount of soluble P leaving reservoir on day (kg P)*
- real \*8 **resnh3o**  
*amount of ammonia leaving reservoir on day (kg N)*
- real \*8 **bactminlp**  
*Threshold detection level for less persistent bacteria. When bacteria levels drop to this amount the model considers bacteria in the soil to be insignificant and sets the levels to zero (cfu/m<sup>2</sup>)*
- real \*8 **bactminp**  
*Threshold detection level for persistent bacteria. When bacteria levels drop to this amount the model considers bacteria in the soil to be insignificant and sets the levels to zero (cfu/m<sup>2</sup>)*
- real \*8 **trnsrch**  
*fraction of transmission losses from main channel that enter deep aquifer*
- real \*8 **wp20p\_plt**  
*overall rate change for persistent bacteria on foliage (1/day)*
- real \*8 **potsedo**  
*sediment leaving pothole to main channel from HRU on day (metric tons/ha)*
- real \*8 **pest\_sol**
- real \*8 **bact\_swf**  
*fraction of manure containing active colony forming units (cfu)*
- real \*8 **bactmx**  
*bacteria percolation coefficient. Ratio of solution bacteria in surface layer to solution bacteria in percolate*
- real \*8 **cncoef**  
*plant ET curve number coefficient*
- real \*8 **wp20lp\_plt**  
*overall rate change for less persistent bacteria on foliage (1/day)*
- real \*8 **cdn\_bsn**
- real \*8 **sdnco\_bsn**
- real \*8 **cn\_froz**  
*drainage coefficient (mm day<sup>-1</sup>)*
- real \*8 **dorm\_hr**  
*time threshold used to define dormant (hours)*
- real \*8 **smxco**  
*adjustment factor for max curve number s factor (0-1)*
- real \*8 **tb\_adj**  
*adjustment factor for subdaily unit hydrograph basetime*
- real \*8 **chla\_subco**  
*regional adjustment on sub chla\_a loading (fraction)*
- real \*8 **depimp\_bsn**  
*depth to impervious layer. Used to model perched water tables in all HRUs in watershed (mm)*
- real \*8 **ddrain\_bsn**  
*depth to the sub-surface drain (mm)*

- real \*8 **rch\_san**
- real \*8 **rch\_sil**
- real \*8 **rch\_cla**
- real \*8 **rch\_sag**
- real \*8 **rch\_lag**
- real \*8 **rch\_gra**
- real \*8 **hlife\_ngw\_bsn**  
*Half-life of nitrogen in groundwater? (days)*
- real \*8 **ch\_opco\_bsn**
- real \*8 **ch\_onco\_bsn**
- real \*8 **decr\_min**  
*Minimum daily residue decay.*
- real \*8 **rcn\_sub\_bsn**  
*Concentration of nitrogen in the rainfall (mg/kg)*
- real \*8 **bc1\_bsn**
- real \*8 **bc2\_bsn**
- real \*8 **bc3\_bsn**
- real \*8 **bc4\_bsn**
- real \*8 **anion\_excl\_bsn**
- real \*8, dimension(:), allocatable **wat\_tbl**  
*water table based on depth from soil surface (mm)*
- real \*8, dimension(:, :), allocatable **vwt**
- real \*8 **re\_bsn**  
*Effective radius of drains (range 3.0 - 40.0) (mm)*
- real \*8 **sdrain\_bsn**  
*Distance bewtween two drain or tile tubes (range 7600.0 - 30000.0) (mm)*
- real \*8 **sstmaxd\_bsn**
- real \*8 **drain\_co\_bsn**  
*Drainage coefficient (range 10.0 - 51.0) (mm-day-1)*
- real \*8 **latksatf\_bsn**  
*Multiplication factor to determine lateral ksat from SWAT ksat input value for HRU (range 0.01 - 4.0)*
- real \*8 **pc\_bsn**  
*Pump capacity (def val = 1.042 mm h-1 or 25 mm day-1) (mm h-1)*
- integer **i\_subhw**
- integer **imgt**
- integer **iwtr**
- integer **mo\_atmo**
- integer **mo\_atmo1**
- integer **iy\_r\_atmo1**
- integer **matmo**
- integer **mch**  
*maximum number of channels*
- integer **mcr**  
*maximum number of crops grown per year*
- integer **mcrdb**  
*maximum number of crops/landcover in database file (crop.dat)*
- integer **mfdb**  
*maximum number of fertilizers in fert.dat*
- integer **mhru**  
*maximum number of HRUs in watershed*
- integer **mhyd**  
*maximum number of hydrograph nodes*

- integer **mpdb**  
*maximum number of pesticides in pest.dat*
- integer **mrq**  
*maximum number of rainfall/temp gages (none)*
- integer **mqr**  
*maximum number of grazings per year*
- integer **mnr**  
*maximum number of years of rotation*
- integer **myr**  
*maximum number of years of simulation*
- integer **isubwq**  
*subbasin water quality code*  
*0 do not calculate algae/CBOD 1 calculate algae/CBOD drainmod tile equations*
- integer **ffcst**
- integer **isproj**  
*special project code (none):*  
*1 test rewind (run simulation twice)*
- integer **nbyr**  
*number of calendar years simulated (none)*
- integer **irte**  
*water routing method (none):*  
*0 variable storage method*  
*1 Muskingum method*
- integer **nrch**  
*number of reaches in watershed (none)*
- integer **nres**  
*total number of reservoirs in watershed (none)*
- integer **nhru**  
*number of last HRU in previous subbasin or*  
*number of HRUs in watershed (none)*
- integer **i\_mo**  
*current month being simulated or month of next day of simulation (none)*
- integer **immo**  
*current cumulative month of simulation (none)*
- integer **wndsim**  
*wind speed input code (noen)*  
*1 measured data read for each subbasin*  
*2 data simulated for each subbasin*
- integer **ihout**  
*variable to hold value for ihouts(:) (none)*
- integer **inum2**  
*variable to hold value for inum2s(:) (none)*
- integer **inum3**  
*variable to hold value for inum3s(:) (none)*
- integer **inum4**  
*variable to hold value for inum4s(:) (none)*
- integer **icfac**  
*icfac = 0 for C-factor calculation using Cmin (as described in manual)*  
*= 1 for new C-factor calculation from RUSLE (no minimum needed)*
- integer **inum5**
- integer **inum6**
- integer **inum7**

- integer **inum8**
- integer **mrech**  
*maximum number of rechour files*
- integer **nrgage**  
*number of raingage files (none)*
- integer **nrgfil**  
*number of rain gages per file (none)*
- integer **nrtot**  
*total number of rain gages (none)*
- integer **ntgage**  
*number of temperature gage files (none)*
- integer **ntgfil**  
*number of temperature gages per file (none)*
- integer **nttot**  
*total number of temperature gages (none)*
- integer **tmpsim**  
*temperature input code (none)*  
*1 measured data read for each subbasin*  
*2 data simulated for each subbasin*
- integer **icrk**  
*crack flow code*  
*1: simulate crack flow in watershed*
- integer **irtpest**  
*number of pesticide to be routed through the watershed. Redefined to the sequence number of pesticide in NPNO(:)*  
*which is to be routed through the watershed (none)*
- integer **igropt**  
*Qual2E option for calculating the local specific growth rate of algae*  
*1: multiplicative.*
- integer **npmx**  
*number of different pesticides used in the simulation (none)*
- integer **curyr**  
*current year in simulation (sequence) (none)*
- integer **itdrn**  
*tile drainage equations flag/code*  
*1 simulate tile flow using subroutine drains(wt\_shall)*  
*0 simulate tile flow using subroutine origtile(wt\_shall,d)*
- integer **iwtdn**  
*water table depth algorithms flag/code*  
*1 simulate wt\_shall using subroutine new water table depth routine*  
*0 simulate wt\_shall using subroutine original water table depth routine*
- integer **ismax**  
*maximum depressional storage selection flag/code (none)*  
*0 = static depressional storage (stmaxd) read from .bsn for the global value or .sdr for specific HRUs*  
*1 = dynamic storage (stmaxd) based on random roughness, tillage and cumulative rainfall intensity by [depstor.f90](#)*
- integer **iroutunit**  
*not being implemented in this version drainmod tile equations*
- integer **ires\_nut**
- integer **iclb**  
*auto-calibration flag*
- integer **mrecc**  
*maximum number of reccnst files*
- integer **mrecd**  
*maximum number of recday files*

- integer **mrecm**  
*maximum number of recmon files*
- integer **mtil**  
*max number of tillage types in till.dat*
- integer **mudb**  
*maximum number of urban land types in urban.dat*
- integer **idist**  
*rainfall distribution code*  
*0 for skewed normal dist*  
*1 for mixed exponential distribution*
- integer **mrecy**  
*maximum number of recyear files*
- integer **nyskip**  
*number of years to skip output summarization and printing (none)*
- integer **slrsim**  
*solar radiation input code (none)*  
*1 measured data read for each subbasin*  
*2 data simulated for each subbasin*
- integer **ideg**  
*channel degradation code*  
*0: do not compute channel degradation*  
*1: compute channel degradation (downcutting and widening)*
- integer **ievent**  
*rainfall/runoff code (none)*  
*0 daily rainfall/curve number technique 1 daily rainfall/curve number technique/ daily routing 2 sub-daily rainfall /↔*  
*Green&Ampt technique/ daily routing 3 sub-daily rainfall /Green&Ampt technique/ hourly routing*
- integer **ipet**  
*code for potential ET method (none)*  
*0 Priestley-Taylor method*  
*1 Penman/Monteith method*  
*2 Hargreaves method*  
*3 read in daily potential ET data*
- integer **iopera**
- integer **idaf**  
*beginning day of simulation (julian date)*
- integer **idal**  
*ending day of simulation (julian date)*
- integer **rhsim**  
*relative humidity input code (none)*  
*1 measured data read for each subbasin*  
*2 data simulated for each subbasin*
- integer **leapyr**  
*leap year flag (none)*  
*0 leap year*  
*1 regular year*
- integer **id1**  
*first day of simulation in current year (julian date)*
- integer **mo\_chk**  
*check for month being simulated; when mo\_chk differs from mo, monthly output is printed (none)*
- integer **nhtot**  
*total number of relative humidity records in file*
- integer **nstot**  
*total number of solar radiation records in file (none)*
- integer **nwtot**

- total number of wind speed records in file*
- integer **ifirsts**
  - solar radiation data search code (none)*
  - 0 first day of solar radiation data located in file*
  - 1 first day of solar radiation data not located in file*
- integer **ifirsth**
  - relative humidity data search code (none)*
  - 0 first day of relative humidity data located in file*
  - 1 first day of relative humidity data not located in file*
- integer **ifirstw**
  - wind speed data search code (none)*
  - 0 first day of wind speed data located in file*
  - 1 first day of wind speed data not located in file*
- integer **ilog**
  - streamflow print code (none)*
  - 0 print streamflow in reach*
  - 1 print Log10 streamflow in reach*
- integer **itotr**
  - number of output variables printed (output.rch)*
- integer **iy**
  - current year of simulation (year)*
- integer **iwq**
  - stream water quality code*
  - 0 do not model stream water quality*
  - 1 model stream water quality (QUAL2E & pesticide transformations)*
- integer **iskip**
  - flag for calculations performed only for the first year of simulation (none)*
- integer **ifirstpet**
  - potential ET data search code (none)*
  - 0 first day of potential ET data located in file*
  - 1 first day of potential ET data not located in file*
- integer **iprp**
  - print code for output.pst file*
  - 0 do not print pesticide output*
  - 1 print pesticide output*
- integer **itotb**
  - number of output variables printed (output.sub)*
- integer **itots**
  - number of output variables printed (output.hru)*
- integer **ith**
  - number of HRUs printed (output.hru/output.wtr)*
- integer **pcpsim**
  - rainfall input code (none)*
  - 1 measured data read for each subbasin*
  - 2 data simulated for each subbasin*
- integer **nd\_30**
- integer **iphr**
- integer **isto**
- integer **isol**
- integer **fcstcycles**
  - number of times forecast period is simulated (using different weather generator seeds each time)*
- integer **fcstday**
  - beginning date of forecast period (julian date)*
- integer **fcstyr**

- beginning year of forecast period*
- integer **iscen**
  - scenarios counter*
- integer **subtot**
  - number of subbasins in watershed (none)*
- integer **ogen**
  - random number generator seed code (none)*
- integer **mlyr**
  - maximum number of soil layers*
- integer **mpst**
  - max number of pesticides used in wshed*
- integer **mres**
  - maximum number of reservoirs*
- integer **msub**
  - maximum number of subbasins*
- integer **igen**
  - random number generator seed code (none):*
  - 0: use default numbers*
  - 1: generate new numbers in every simulation*
- integer **iprint**
  - print code (none): 0=monthly, 1=daily, 2=annually*
- integer **iida**
  - day being simulated (current julian date) (julian date)*
- integer **icn**
  - CN method flag (for testing alternative method):*
  - 0 use traditional SWAT method which bases CN on soil moisture*
  - 1 use alternative method which bases CN on plant ET*
  - 2 use traditional SWAT method which bases CN on soil moisture but rention is adjusted for mildly-sloped tiled-drained watersheds.*
- integer **ised\_det**
  - max half-hour rainfall fraction calc option:*
  - 0 generate max half-hour rainfall fraction from triangular distribution*
  - 1 use monthly mean max half-hour rainfall fraction*
- integer **fcstcnt**
- integer **idtil**
- integer, dimension(100) **ida\_lup**
- integer, dimension(100) **iy\_r\_lup**
- integer **no\_lup**
- integer **nostep**
- character(len=13) **rhfile**
  - relative humidity file name (.hmd)*
- character(len=13) **slrfile**
  - solar radiation file name (.slr)*
- character(len=13) **wndfile**
  - wind speed file name (.wnd)*
- character(len=13) **petfile**
  - potential ET file name (.pet)*
- character(len=13) **atmofile**
- character(len=13) **septdb**
  - name of septic tank database file (septwq1.dat)*
- integer, dimension(9) **idg**
  - array location of random number seed used for a given process*
- integer, dimension(:), allocatable **ifirsthr**

- measured data search code (none)*
  - 0 first day of measured data located in file*
  - 1 first day of measured data not located in file*
- integer, dimension(:), allocatable **ifistr**
  - measured data search code (none)*
  - 0 first day of measured data located in file*
  - 1 first day of measured data not located in file*
- integer, dimension(8) **values**
  - values(1): year simulation is performed*
  - values(2): month simulation is performed*
  - values(3): day in month simulation is performed*
  - values(4): time difference with respect to Coordinated Universal Time (ie Greenwich Mean Time)*
  - values(5): hour simulation is performed*
  - values(6): minute simulation is performed*
  - values(7): second simulation is performed*
  - values(8): millisecond simulation is performed*
- integer, dimension(13) **ndays**
  - julian date for last day of preceding month (where the array location is the number of the month). The dates are for leap years (julian date)*
- integer **mapex**
- real \*8, dimension(:), allocatable **hi\_targ**
  - harvest index target of cover defined at planting ((kg/ha)/(kg/ha))*
- real \*8, dimension(:), allocatable **bio\_targ**
  - biomass target (kg/ha)*
- real \*8, dimension(:), allocatable **tnyld**
  - modifier for autofertilization target nitrogen content for plant (kg N/kg yield)*
- integer, dimension(:), allocatable **ifirsta**
- integer, dimension(100) **mo\_transb**
- integer, dimension(100) **mo\_transe**
- integer, dimension(100) **ih\_tran**
- integer **msdb**
  - maximum number of sept wq data database (none)*
- integer **iseptic**
- real \*8, dimension(:), allocatable **sptqs**
  - flow rate of the septic tank effluent per capita (m3/d)*
- real \*8, dimension(:), allocatable **sptbodconcs**
  - Biological Oxygen Demand of the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptssconcs**
  - concentration of total suspended solid in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptnh4concs**
  - concentration of total phosphorus of the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptno3concs**
  - concentration of nitrate in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptno2concs**
  - concentration of nitrite in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptorgnconcs**
  - concentration of organic nitrogen in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptminps**
  - concentration of mineral phosphorus in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptorgps**
  - concentration of organic phosphorus in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable **sptfcolis**
  - concentration of the facel caliform in the septic tank effluent (cfu/100ml)*
- real \*8, dimension(:), allocatable **failyr**



- real \*8, dimension(:), allocatable **qstemm**
- real \*8, dimension(:), allocatable **bio\_bod**  
*BOD concentration in biozone (kg/ha)*
- real \*8, dimension(:), allocatable **biom**  
*biomass of live bacteria in biozone (kg/ha)*
- real \*8, dimension(:), allocatable **rbiom**  
*daily change in biomass of live bacteria (kg/ha)*
- real \*8, dimension(:), allocatable **fcoli**  
*concentration of the fecal coliform in the biozone septic tank effluent (cfu/100ml)*
- real \*8, dimension(:), allocatable **bz\_perc**
- real \*8, dimension(:), allocatable **plqm**  
*plaque in biozone (kg/ha)*
- real \*8, dimension(:), allocatable **bz\_area**
- real \*8, dimension(:), allocatable **bz\_z**  
*depth of biozone layer (mm)*
- real \*8, dimension(:), allocatable **bz\_thk**  
*thickness of biozone (mm)*
- real \*8, dimension(:), allocatable **bio\_bd**  
*density of biomass (kg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **cmup\_kgh**  
*current soil carbon for first soil layer (kg/ha)*
- real \*8, dimension(:), allocatable **cmtot\_kgh**  
*current soil carbon integrated - aggregating (kg/ha)*
- real \*8, dimension(:), allocatable **coeff\_denitr**  
*denitrification rate coefficient (none)*
- real \*8, dimension(:), allocatable **coeff\_bod\_dc**  
*BOD decay rate coefficient (m<sup>3</sup>/day)*
- real \*8, dimension(:), allocatable **coeff\_bod\_conv**  
*BOD to live bacteria biomass conversion factor (none)*
- real \*8, dimension(:), allocatable **coeff\_fc1**  
*field capacity calibration parameter 1 (none)*
- real \*8, dimension(:), allocatable **coeff\_fc2**  
*field capacity calibration parameter 2 (none)*
- real \*8, dimension(:), allocatable **coeff\_fecal**  
*fecal coliform bacteria decay rate coefficient (m<sup>3</sup>/day)*
- real \*8, dimension(:), allocatable **coeff\_mrt**  
*mortality rate coefficient (none)*
- real \*8, dimension(:), allocatable **coeff\_nitr**  
*nitrification rate coefficient (none)*
- real \*8, dimension(:), allocatable **coeff\_plq**  
*conversion factor for plaque from TDS (none)*
- real \*8, dimension(:), allocatable **coeff\_rsp**  
*respiration rate coefficient (none)*
- real \*8, dimension(:), allocatable **coeff\_slg1**  
*slough-off calibration parameter (none)*
- real \*8, dimension(:), allocatable **coeff\_slg2**  
*slough-off calibration parameter (none)*
- real \*8, dimension(:), allocatable **coeff\_pdistrb**
- real \*8, dimension(:), allocatable **coeff\_solpslp**
- real \*8, dimension(:), allocatable **coeff\_solpintc**
- real \*8, dimension(:), allocatable **coeff\_psorpmax**

- integer, dimension(:), allocatable [isep\\_typ](#)  
*septic system type (none)*
- integer, dimension(:), allocatable [i\\_sep](#)  
*soil layer where biozone exists (none)*
- integer, dimension(:), allocatable [isep\\_opt](#)  
*septic system operation flag (1=active, 2=failing, 3 or 0=not operated) (none)*
- integer, dimension(:), allocatable **sep\_tsincefail**
- integer, dimension(:), allocatable **isep\_tfail**
- integer, dimension(:), allocatable **isep\_iyr**
- real \*8, dimension(:), allocatable **sol\_sumno3**
- real \*8, dimension(:), allocatable **sol\_sumsolp**
- real \*8, dimension(:), allocatable **strsw\_sum**
- real \*8, dimension(:), allocatable **strstmp\_sum**
- real \*8, dimension(:), allocatable **strsn\_sum**
- real \*8, dimension(:), allocatable **strsp\_sum**
- real \*8, dimension(:), allocatable **strsa\_sum**
- real \*8, dimension(:), allocatable **pot\_seep**
- real \*8, dimension(:), allocatable [pot\\_solp](#)  
*soluble P loss rate in the pothole (.01 - 0.5) (1/d)*
- real \*8, dimension(:), allocatable [pot\\_orgp](#)  
*amount of organic P in pothole water body (kg P)*
- real \*8, dimension(:), allocatable [pot\\_orgn](#)  
*amount of organic N in pothole water body (kg N)*
- real \*8, dimension(:), allocatable [pot\\_mps](#)  
*amount of stable mineral pool P in pothole water body (kg N)*
- real \*8, dimension(:), allocatable [pot\\_mpa](#)  
*amount of active mineral pool P in pothole water body (kg N)*
- real \*8, dimension(:), allocatable **tile\_solpo**
- integer [ia\\_b](#)  
*print ascii or binary files (none)*
- integer [ihumus](#)  
*ihumus = 0 do no print file*  
*ihumus = 1 print output.wql*
- integer **itemp**
- integer **isnow**
- integer, dimension(46) [ipdvar](#)  
*output variable codes for output.rch file (none)*
- integer, dimension([mhruo](#)) [ipdvas](#)  
*output variable codes for output.hru file (none)*
- integer, dimension([msubo](#)) [ipdvab](#)  
*output variable codes for output.sub file (none)*
- integer, dimension(:), allocatable [ipdhru](#)  
*HRUs whose output information will be printed to the output.hru and output.wtr files.*
- real \*8, dimension([mstdo](#)) [wshddayo](#)  
*watershed daily output array (varies)*  
*wshddayo(1) average amountof precipitation in watershed for the day (mm H2O)*  
*wshddayo(3) surface runoff in watershed for day (mm H2O)*  
*wshddayo(4) lateral flow contribution to streamflow in watershed for day (mm H2O)*  
*wshddayo(5) water percolation past bottom of soil profile in watershed for day (mm H2O)*  
*wshddayo(6) water yield to streamflow from HRUs in watershed for day (mm H2O)*  
*wshddayo(7) actual evapotranspiration in watershed for day (mm H2O)*  
*wshddayo(8) average maximum temperature in watershed for the day (deg C)*  
*wshddayo(9) average minimum temperature in watershed for the day (deg C)*  
*wshddayo(11) net change in sediment of reservoirs in watershed for day (metric tons)*

*wshddayo(12)* sediment yield from HRUs in watershed for day (metric tons or metric tons/ha)  
*wshddayo(13)* sediment loading to ponds in watershed for day (metric tons)  
*wshddayo(14)* sediment loading from ponds in watershed for day (metric tons)  
*wshddayo(15)* net change in sediment level in ponds in watershed for day (metric tons)  
*wshddayo(16)* sediment loading to wetlands for day in watershed (metric tons)  
*wshddayo(17)* sediment loading to main channels from wetlands for day in watershed (metric tons)  
*wshddayo(18)* net change in sediment in wetlands for day in watershed (metric tons)  
*wshddayo(19)* evaporation from ponds in watershed for day ( $m^3 H_2O$ )  
*wshddayo(20)* seepage from ponds in watershed for day ( $m^3 H_2O$ )  
*wshddayo(21)* precipitation on ponds in watershed for day ( $m^3 H_2O$ )  
*wshddayo(22)* volume of water entering ponds in watershed for day ( $m^3 H_2O$ )  
*wshddayo(23)* volume of water leaving ponds in watershed for day ( $m^3 H_2O$ )  
*wshddayo(24)* evaporation from wetlands for day in watershed ( $m^3 H_2O$ )  
*wshddayo(25)* seepage from wetlands for day in watershed ( $m^3 H_2O$ )  
*wshddayo(26)* precipitation on wetlands for day in watershed ( $m^3 H_2O$ )  
*wshddayo(27)* volume of water entering wetlands on day in watershed ( $m^3 H_2O$ )  
*wshddayo(28)* volume of water leaving wetlands on day in watershed ( $m^3 H_2O$ )  
*wshddayo(33)* net change in water volume of ponds in watershed for day ( $m^3 H_2O$ )  
*wshddayo(34)* net change in water volume of reservoirs in watershed for day ( $m^3 H_2O$ )  
*wshddayo(35)* amount of water stored in soil profile in watershed at end of day (mm H<sub>2</sub>O)  
*wshddayo(36)* snow melt in watershed for day (mm H<sub>2</sub>O)  
*wshddayo(37)* sublimation in watershed for day (mm H<sub>2</sub>O)  
*wshddayo(38)* average amount of tributary channel transmission losses in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(39)* freezing rain/snow fall in watershed for day (mm H<sub>2</sub>O)  
*wshddayo(40)* organic N loading to stream in watershed for day (kg N/ha)  
*wshddayo(41)* organic P loading to stream in watershed for day (kg P/ha)  
*wshddayo(42)* nitrate loading to stream in surface runoff in watershed for day (kg N/ha)  
*wshddayo(43)* soluble P loading to stream in watershed for day (kg P/ha)  
*wshddayo(44)* plant uptake of N in watershed for day (kg N/ha)  
*wshddayo(45)* nitrate loading to stream in lateral flow in watershed for day (kg N/ha)  
*wshddayo(46)* nitrate percolation past bottom of soil profile in watershed for day (kg N/ha)  
*wshddayo(104)* groundwater contribution to stream in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(105)* amount of water moving from shallow aquifer to plants/soil profile in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(106)* deep aquifer recharge in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(107)* total amount of water entering both aquifers in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(108)* potential evapotranspiration in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(109)* drainage tile flow contribution to stream in watershed on day (mm H<sub>2</sub>O)  
*wshddayo(110)* NO<sub>3</sub> yield (gwq) (kg/ha)  
*wshddayo(111)* NO<sub>3</sub> yield (tile) (mm H<sub>2</sub>O)

- real \*8, dimension(**mstdo**) **wshdmono**

watershed monthly output array (see definitions for *wshddayo* array elements) (varies)  
*wshdmono(1)* average amount of precipitation in watershed for the month (mm H<sub>2</sub>O)  
*wshdmono(3)* surface runoff in watershed for month (mm H<sub>2</sub>O)  
*wshdmono(4)* lateral flow contribution to streamflow in watershed for month (mm H<sub>2</sub>O)  
*wshdmono(5)* water percolation past bottom of soil profile in watershed for month (mm H<sub>2</sub>O)  
*wshdmono(6)* water yield to streamflow from HRUs in watershed for month (mm H<sub>2</sub>O)  
*wshdmono(7)* actual evapotranspiration in watershed for month (mm H<sub>2</sub>O)  
*wshdmono(8)* average maximum temperature in watershed for the month (deg C)  
*wshdmono(9)* average minimum temperature in watershed for the month (deg C)  
*wshdmono(12)* sediment yield from HRUs in watershed for the month (metric tons)  
*wshdmono(39)* freezing rain/snow fall in watershed for the month (mm H<sub>2</sub>O)  
*wshdmono(40)* organic N loading to stream in watershed for the month (kg N/ha)  
*wshdmono(41)* organic P loading to stream in watershed for the month (kg P/ha)  
*wshdmono(42)* nitrate loading to stream in surface runoff in watershed for the month (kg N/ha)  
*wshdmono(43)* soluble P loading to stream in watershed for the month (kg P/ha)  
*wshdmono(44)* plant uptake of N in watershed for the month (kg N/ha)  
*wshdmono(45)* nitrate loading to stream in lateral flow in watershed for the month (kg N/ha)  
*wshdmono(46)* nitrate percolation past bottom of soil profile in watershed for the month (kg N/ha)  
*wshdmono(104)* groundwater contribution to stream in watershed for the month (mm H<sub>2</sub>O)  
*wshdmono(108)* potential evapotranspiration in watershed for the month (mm H<sub>2</sub>O)  
*wshdmono(109)* drainage tile flow contribution to stream in watershed for the month (mm H<sub>2</sub>O)

- real \*8, dimension(**mstdo**) **wshdyro**

watershed annual output array (varies)

- wshdyro(1)* average amount of precipitation in watershed for the year (mm H2O)
- wshdyro(3)* surface runoff in watershed for year (mm H2O)
- wshdyro(4)* lateral flow contribution to streamflow in watershed for year (mm H2O)
- wshdyro(5)* water percolation past bottom of soil profile in watershed for year (mm H2O)
- wshdyro(6)* water yield to streamflow from HRUs in watershed for year (mm H2O)
- wshdyro(7)* actual evapotranspiration in watershed for year (mm H2O)
- wshdyro(8)* average maximum temperature in watershed for the year (deg C)
- wshdyro(9)* average minimum temperature in watershed for the year (deg C)
- wshdaao(12)* sediment yield from HRUs in watershed for the year (metric tons)
- wshdyro(40)* organic N loading to stream in watershed for the year (kg N/ha)
- wshdyro(41)* organic P loading to stream in watershed for the year (kg P/ha)
- wshdyro(42)* nitrate loading to stream in surface runoff in watershed for the year (kg N/ha)
- wshdyro(43)* soluble P loading to stream in watershed for the year (kg P/ha)
- wshdyro(44)* plant uptake of N in watershed for the year
- wshdyro(45)* nitrate loading to stream in lateral flow in watershed for the year (kg N/ha)
- wshdyro(46)* nitrate percolation past bottom of soil profile in watershed for the year (kg N/ha)
- wshdyro(104)* groundwater contribution to stream in watershed for the year (mm H2O)
- wshdyro(108)* potential evapotranspiration in watershed for the year (mm H2O)
- wshdyro(109)* drainage tile flow contribution to stream in watershed for the year (mm H2O)
- real \*8, dimension(16) **fcstaa0**
- real \*8, dimension([mstdo](#)) **wshdaao**
  - watershed average annual output array (varies)*
  - wshdaao(1)* precipitation in watershed (mm H2O)
  - wshdaao(3)* surface runoff loading to main channel in watershed (mm H2O)
  - wshdaao(4)* lateral flow loading to main channel in watershed (mm H2O)
  - wshdaao(5)* percolation of water out of root zone in watershed (mm H2O)
  - wshdaao(6)* water yield to streamflow from HRUs in watershed for simulation (mm H2O)
  - wshdaao(7)* actual evapotranspiration in watershed (mm H2O)
  - wshdaao(11)* net change in sediment of reservoirs in watershed during simulation (metric tons/ha)
  - wshdaao(12)* sediment yield from HRUs in watershed for the simulation (metric tons/ha)
  - wshdaao(13)* sediment loading to ponds in watershed during simulation (metric tons/ha)
  - wshdaao(14)* sediment loading from ponds in watershed during simulation (metric tons/ha)
  - wshdaao(15)* net change in sediment level in ponds in watershed during simulation (metric tons/ha)
  - wshdaao(19)* evaporation from ponds in watershed (m<sup>3</sup> H2O)
  - wshdaao(19)* evaporation from ponds in watershed during simulation (mm H2O)
  - wshdaao(20)* seepage from ponds in watershed during simulation (mm H2O)
  - wshdaao(21)* precipitation on ponds in watershed during simulation (mm H2O)
  - wshdaao(22)* volume of water entering ponds in watershed during simulation (mm H2O)
  - wshdaao(23)* volume of water leaving ponds in watershed during simulation (mm H2O)
  - wshdaao(33)* net change in water volume of ponds in watershed during simulation (mm H2O)
  - wshdaao(34)* net change in water volume of reservoirs in watershed during simulation (mm H2O)
  - wshdaao(36)* snow melt in watershed for simulation (mm H2O)
  - wshdaao(38)* average amount of tributary channel transmission losses in watershed during simulation (mm H2O)
  - wshdaao(39)* freezing rain/snow fall in watershed for the simulation (mm H2O)
  - wshdaao(40)* organic N loading to stream in watershed for the simulation (kg N/ha)
  - wshdaao(41)* organic P loading to stream in watershed for the simulation (kg P/ha)
  - wshdaao(42)* nitrate loading to stream in surface runoff in watershed for the simulation (kg N/ha)
  - wshdaao(43)* soluble P loading to stream in watershed for the simulation (kg P/ha)
  - wshdaao(44)* plant uptake of N in watershed for the simulation (kg N/ha)
  - wshdaao(45)* nitrate loading to stream in lateral flow in watershed for the simulation (kg N/ha)
  - wshdaao(46)* nitrate percolation past bottom of soil profile in watershed for the simulation (kg N/ha)
  - wshdaao(104)* groundwater contribution to stream in watershed for the simulation (shallow aquifer) (mm H2O)
  - wshdaao(105)* amount of water moving from shallow aquifer to plants/soil profile in watershed during simulation (mm H2O)
  - wshdaao(106)* deep aquifer recharge in watershed during simulation (mm H2O)
  - wshdaao(107)* total amount of water entering both aquifers in watershed during simulation (mm H2O)
  - wshdaao(108)* potential evapotranspiration in watershed for the simulation (mm H2O)
  - wshdaao(109)* drainage tile flow contribution to stream in watershed for the simulation (mm H2O)
  - wshdaao(113)* groundwater contribution to stream in watershed for the simulation (deep aquifer) (mm H2O)
- real \*8, dimension(:,.), allocatable **wpstdayo**
  - watershed daily pesticide output array (varies)*
  - wpstdayo(1,:) amount of pesticide type in surface runoff contribution to stream in watershed on day (in solution) (mg pst/ha)*

- wpstdayo(2,:) amount of pesticide type in surface runoff contribution to stream in watershed on day (sorbed to sediment) (mg pst/ha)*
- wpstdayo(3,:) amount of pesticide type leached from soil profile in watershed on day (kg pst/ha)*
- wpstdayo(4,:) amount of pesticide type in lateral flow contribution to stream in watershed on day (kg pst/ha)*
- real \*8, dimension(:,:), allocatable **wpstmono**
- real \*8, dimension(:,:), allocatable **wpstyro**
- real \*8, dimension(:,:), allocatable **bio\_hv**
  - harvested biomass (dry weight) (kg/ha)*
- real \*8, dimension(:,:), allocatable **yldkg**
  - yield (dry weight) by crop type in the HRU (kg/ha)*
- real \*8, dimension(:,:), allocatable **rchmono**
  - reach monthly output array (varies)*
  - rchmono(1,:) flow into reach during month ( $m^3/s$ )*
  - rchmono(2,:) flow out of reach during month ( $m^3/s$ )*
  - rchmono(3,:) sediment transported into reach during month (metric tons)*
  - rchmono(4,:) sediment transported out of reach during month (metric tons)*
  - rchmono(5,:) sediment concentration in outflow during month (mg/L)*
  - rchmono(6,:) organic N transported into reach during month (kg N)*
  - rchmono(7,:) organic N transported out of reach during month (kg N)*
  - rchmono(8,:) organic P transported into reach during month (kg P)*
  - rchmono(9,:) organic P transported out of reach during month (kg P)*
  - rchmono(10,:) evaporation from reach during month ( $m^3/s$ )*
  - rchmono(11,:) transmission losses from reach during month ( $m^3/s$ )*
  - rchmono(12,:) conservative metal #1 transported out of reach during month (kg)*
  - rchmono(13,:) conservative metal #2 transported out of reach during month (kg)*
  - rchmono(14,:) conservative metal #3 transported out of reach during month (kg)*
  - rchmono(15,:) nitrate transported into reach during month (kg N)*
  - rchmono(16,:) nitrate transported out of reach during month (kg N)*
  - rchmono(17,:) soluble P transported into reach during month (kg P)*
  - rchmono(18,:) soluble P transported out of reach during month (kg P)*
  - rchmono(19,:) soluble pesticide transported into reach during month (mg pst)*
  - rchmono(20,:) soluble pesticide transported out of reach during month (mg pst)*
  - rchmono(21,:) sorbed pesticide transported into reach during month (mg pst)*
  - rchmono(22,:) sorbed pesticide transported out of reach during month (mg pst)*
  - rchmono(23,:) amount of pesticide lost through reactions in reach during month (mg pst)*
  - rchmono(24,:) amount of pesticide lost through volatilization from reach during month (mg pst)*
  - rchmono(25,:) amount of pesticide settling out of reach to bed sediment during month (mg pst)*
  - rchmono(26,:) amount of pesticide resuspended from bed sediment to reach during month (mg pst)*
  - rchmono(27,:) amount of pesticide diffusing from reach to bed sediment during month (mg pst)*
  - rchmono(28,:) amount of pesticide in sediment layer lost through reactions during month (mg pst)*
  - rchmono(29,:) amount of pesticide in sediment layer lost through burial during month (mg pst)*
  - rchmono(30,:) chlorophyll-a transported into reach during month (kg chla)*
  - rchmono(31,:) chlorophyll-a transported out of reach during month (kg chla)*
  - rchmono(32,:) ammonia transported into reach during month (kg N)*
  - rchmono(33,:) ammonia transported out of reach during month (kg N)*
  - rchmono(34,:) nitrite transported into reach during month (kg N)*
  - rchmono(35,:) nitrite transported out of reach during month (kg N)*
  - rchmono(36,:) CBOD transported into reach during month (kg O2)*
  - rchmono(37,:) CBOD transported out of reach during month (kg O2)*
  - rchmono(38,:) dissolved oxygen transported into reach during month (kg O2)*
  - rchmono(39,:) dissolved oxygen transported out of reach during month (kg O2)*
  - rchmono(40,:) persistent bacteria transported out of reach during month (kg bact)*
  - rchmono(41,:) less persistent bacteria transported out of reach during month (kg bact)*
  - rchmono(43,:) total N (org N + no3 + no2 + nh4 outs) (kg)*
  - rchmono(44,:) total P (org P + sol p outs) (kg)*
- real \*8, dimension(:,:), allocatable **rchyro**
  - reach annual output array (varies)*
  - rchyro(1,:) flow into reach during year ( $m^3/s$ )*
  - rchyro(2,:) flow out of reach during year ( $m^3/s$ )*
  - rchyro(3,:) sediment transported into reach during year (metric tons)*
  - rchyro(4,:) sediment transported out of reach during year (metric tons)*



- rchyro(5,:)* sediment concentration in outflow during year (mg/L)
- rchyro(6,:)* organic N transported into reach during year (kg N)
- rchyro(7,:)* organic N transported out of reach during year (kg N)
- rchyro(8,:)* organic P transported into reach during year (kg P)
- rchyro(9,:)* organic P transported out of reach during year (kg P)
- rchyro(10,:)* evaporation from reach during year ( $m^3/s$ )
- rchyro(11,:)* transmission losses from reach during year ( $m^3/s$ )
- rchyro(12,:)* conservative metal #1 transported out of reach during year (kg)
- rchyro(13,:)* conservative metal #2 transported out of reach during year (kg)
- rchyro(14,:)* conservative metal #3 transported out of reach during year (kg)
- rchyro(15,:)* nitrate transported into reach during year (kg N)
- rchyro(16,:)* nitrate transported out of reach during year (kg N)
- rchyro(17,:)* soluble P transported into reach during year (kg P)
- rchyro(18,:)* soluble P transported out of reach during year (kg P)
- rchyro(19,:)* soluble pesticide transported into reach during year (mg pst)
- rchyro(20,:)* soluble pesticide transported out of reach during year (mg pst)
- rchyro(21,:)* sorbed pesticide transported into reach during year (mg pst)
- rchyro(22,:)* sorbed pesticide transported out of reach during year (mg pst)
- rchyro(23,:)* amount of pesticide lost through reactions in reach during year!> (mg pst)
- rchyro(24,:)* amount of pesticide lost through volatilization from reach during year (mg pst)
- rchyro(25,:)* amount of pesticide settling out of reach to bed sediment during year (mg pst)
- rchyro(26,:)* amount of pesticide resuspended from bed sediment to reach during year (mg pst)
- rchyro(27,:)* amount of pesticide diffusing from reach to bed sediment during year (mg pst)
- rchyro(28,:)* amount of pesticide in sediment layer lost through reactions during year (mg pst)
- rchyro(29,:)* amount of pesticide in sediment layer lost through burial during year (mg pst)
- rchyro(30,:)* chlorophyll-a transported into reach during year (kg chla)
- rchyro(31,:)* chlorophyll-a transported out of reach during year (kg chla)
- rchyro(32,:)* ammonia transported into reach during year (kg N)
- rchyro(33,:)* ammonia transported out of reach during year (kg N)
- rchyro(34,:)* nitrite transported into reach during year (kg N)
- rchyro(35,:)* nitrite transported out of reach during year (kg N)
- rchyro(36,:)* CBOD transported into reach during year (kg O<sub>2</sub>)
- rchyro(37,:)* CBOD transported out of reach during year (kg O<sub>2</sub>)
- rchyro(38,:)* dissolved oxygen transported into reach during year (kg O<sub>2</sub>)
- rchyro(39,:)* dissolved oxygen transported out of reach during year (kg O<sub>2</sub>)
- rchyro(40,:)* persistent bacteria transported out of reach during year (kg bact)
- rchyro(41,:)* less persistent bacteria transported out of reach during year (kg bact)
- real \*8, dimension(:,:), allocatable **wpstaao**
  - wpstaao(1,:)* amount of pesticide type in surface runoff contribution to stream in watershed (in solution) - average annual (mg pst/ha)
  - wpstaao(2,:)* amount of pesticide type in surface runoff contribution to stream in watershed (sorbed to sediment) -average annual (mg pst/ha)
  - wpstaao(3,:)* amount of pesticide type leached from soil profile in watershed - average annual (kg pst/ha)
  - wpstaao(4,:)* amount of pesticide type in lateral flow contribution to stream in watershed - average annual (kg pst/ha)
- real \*8, dimension(:,:), allocatable **hrumono**
  - HRU* monthly output data array (varies)
  - hrumono(1,:)* precipitation in HRU during month (mm H<sub>2</sub>O)
  - hrumono(2,:)* amount of precipitation falling as freezing rain/snow in HRU during month (mm H<sub>2</sub>O)
  - hrumono(3,:)* amount of snow melt in HRU during month (mm H<sub>2</sub>O)
  - hrumono(4,:)* amount of surface runoff to main channel from HRU during month (ignores impact of transmission losses) (mm H<sub>2</sub>O)
  - hrumono(5,:)* amount of lateral flow contribution to main channel from HRU during month (mm H<sub>2</sub>O)
  - hrumono(6,:)* amount of groundwater flow contribution to main channel from HRU during month (mm H<sub>2</sub>O)
  - hrumono(7,:)* amount of water moving from shallow aquifer to plants or soil profile in HRU during mont (mm H<sub>2</sub>O)h
  - hrumono(8,:)* amount of water recharging deep aquifer in HRU during month (mm H<sub>2</sub>O)
  - hrumono(9,:)* total amount of water entering both aquifers from HRU during month (mm H<sub>2</sub>O)
  - hrumono(10,:)* water yield (total amount of water entering main channel) from HRU during month (mm H<sub>2</sub>O)
  - hrumono(11,:)* amount of water percolating out of the soil profile and into the vadose zone in HRU during month (mm H<sub>2</sub>O)
  - hrumono(12,:)* actual evapotranspiration in HRU during month (mm H<sub>2</sub>O)
  - hrumono(13,:)* amount of transmission losses from tributary channels in HRU for month (mm H<sub>2</sub>O)
  - hrumono(14,:)* sediment yield from HRU for month (metric tons/ha)
  - hrumono(15,:)* actual amount of transpiration that occurs during month in HRU (mm H<sub>2</sub>O)

- humono(16,:)* actual amount of evaporation (from soil) that occurs during month in HRU (mm H<sub>2</sub>O)  
*humono(17,:)* amount of nitrogen applied in continuous fertilizer operation during month in HRU (kg N/ha)  
*humono(18,:)* amount of phosphorus applied in continuous fertilizer operation during month in HRU (kg P/ha)  
*humono(19,:)* amount of surface runoff generated during month in HRU (mm H<sub>2</sub>O)  
*humono(20,:)* CN values during month in HRU (none)  
*humono(21,:)* sum of daily soil water values used to calculate the curve number (mm H<sub>2</sub>O)  
*humono(22,:)* amount of irrigation water applied to HRU during month (mm H<sub>2</sub>O)  
*humono(23,:)* amount of water removed from shallow aquifer in HRU for irrigation during month (mm H<sub>2</sub>O)  
*humono(24,:)* amount of water removed from deep aquifer in HRU for irrigation during month (mm H<sub>2</sub>O)  
*humono(25,:)* potential evapotranspiration in HRU during month (mm H<sub>2</sub>O)  
*humono(26,:)* monthly amount of N (organic & mineral) applied in HRU during grazing (kg N/ha)  
*humono(27,:)* monthly amount of P (organic & mineral) applied in HRU during grazing (kg P/ha)  
*humono(28,:)* monthly amount of N (organic & mineral) auto-applied in HRU (kg N/ha)  
*humono(29,:)* monthly amount of P (organic & mineral) auto-applied in HRU (kg P/ha)  
*humono(30,:)* sum of daily soil temperature values (deg C) *humono(31,:)* water stress days in HRU during month (stress days)  
*humono(32,:)* temperature stress days in HRU during month (stress days)  
*humono(33,:)* nitrogen stress days in HRU during month (stress days)  
*humono(34,:)* phosphorus stress days in HRU during month (stress days)  
*humono(35,:)* organic nitrogen in surface runoff in HRU during month (kg N/ha)  
*humono(36,:)* organic phosphorus in surface runoff in HRU during month (kg P/ha)  
*humono(37,:)* nitrate in surface runoff in HRU during month (kg N/ha)  
*humono(38,:)* nitrate in lateral flow in HRU during month (kg N/ha)  
*humono(39,:)* soluble phosphorus in surface runoff in HRU during month (kg P/ha)  
*humono(40,:)* amount of nitrogen removed from soil by plant uptake in HRU during month (kg N/ha)  
*humono(41,:)* nitrate percolating past bottom of soil profile in HRU during month (kg N/ha)  
*humono(42,:)* amount of phosphorus removed from soil by plant uptake in HRU during month (kg P/ha)  
*humono(43,:)* amount of phosphorus moving from labile mineral to active mineral pool in HRU during month (kg P/ha)  
*humono(44,:)* amount of phosphorus moving from active mineral to stable mineral pool in HRU during month (kg P/ha)  
*humono(45,:)* amount of nitrogen applied to HRU in fertilizer and grazing operations during month (kg N/ha)  
*humono(46,:)* amount of phosphorus applied to HRU in fertilizer and grazing operations during month (kg P/ha)  
*humono(47,:)* amount of nitrogen added to soil by fixation in HRU during month (kg N/ha)  
*humono(48,:)* amount of nitrogen lost by denitrification in HRU during month (kg N/ha)  
*humono(49,:)* amount of nitrogen moving from active organic to nitrate pool in HRU during month (kg N/ha)  
*humono(50,:)* amount of nitrogen moving from active organic to stable organic pool in HRU during month (kg N/ha)  
*humono(51,:)* amount of phosphorus moving from organic to labile mineral pool in HRU during month (kg P/ha)  
*humono(52,:)* amount of nitrogen moving from fresh organic to nitrate and active organic pools in HRU during month (kg N/ha)  
*humono(53,:)* amount of phosphorus moving from fresh organic to the labile mineral and organic pools in HRU during month (kg P/ha)  
*humono(54,:)* amount of nitrogen added to soil in rain (kg N/ha)  
*humono(61,:)* daily soil loss predicted with USLE equation (metric tons/ha)  
*humono(62,:)* drainage tile flow contribution to main channel from HRU in month (mm H<sub>2</sub>O)  
*humono(63,:)* less persistent bacteria transported to main channel from HRU during month (bacteria/ha)  
*humono(64,:)* persistent bacteria transported to main channel from HRU during month (bacteria/ha)  
*humono(65,:)* nitrate loading from groundwater in HRU to main channel during month (kg N/ha)  
*humono(66,:)* soluble P loading from groundwater in HRU to main channel during month (kg P/ha)  
*humono(67,:)* loading of mineral P attached to sediment in HRU to main channel during month (kg P/ha)
- real \*8, dimension(:,:), allocatable **rchdy**  
 daily reach output array (varies)  
*rchdy(1,:)* flow into reach on day (m<sup>3</sup>/s)  
*rchdy(2,:)* flow out of reach on day (m<sup>3</sup>/s)  
*rchdy(3,:)* evaporation from reach on day (m<sup>3</sup>/s)  
*rchdy(4,:)* transmission losses from reach on day (m<sup>3</sup>/s)  
*rchdy(5,:)* sediment transported into reach on day (metric tons)  
*rchdy(6,:)* sediment transported out of reach on day (metric tons)  
*rchdy(7,:)* sediment concentration in outflow (mg/L)  
*rchdy(8,:)* organic N transported into reach on day (kg N)  
*rchdy(9,:)* organic N transported out of reach on day (kg N)  
*rchdy(10,:)* organic P transported into reach on day (kg P)  
*rchdy(11,:)* organic P transported out of reach on day (kg P)

- rchdy(12,:) nitrate transported into reach on day (kg N)*
- rchdy(13,:) nitrate transported out of reach on day (kg N)*
- rchdy(14,:) ammonia transported into reach on day (kg N)*
- rchdy(15,:) ammonia transported out of reach on day (kg N)*
- rchdy(16,:) nitrite transported into reach on day (kg N)*
- rchdy(17,:) nitrite transported out of reach on day (kg N)*
- rchdy(18,:) soluble P transported into reach on day (kg P)*
- rchdy(19,:) soluble P transported out of reach on day (kg P)*
- rchdy(20,:) chlorophyll-a transported into reach on day (kg chl-a)*
- rchdy(21,:) chlorophyll-a transported out of reach on day (kg chl-a)*
- rchdy(22,:) CBOD transported into reach on day (kg O2)*
- rchdy(23,:) CBOD transported out of reach on day (kg O2)*
- rchdy(24,:) dissolved oxygen transported into reach on day (kg O2)*
- rchdy(25,:) dissolved oxygen transported out of reach on day (kg O2)*
- rchdy(26,:) soluble pesticide transported into reach on day (mg pst)*
- rchdy(27,:) soluble pesticide transported out of reach on day (mg pst)*
- rchdy(28,:) sorbed pesticide transported into reach on day (mg pst)*
- rchdy(29,:) sorbed pesticide transported out of reach on day (mg pst)*
- rchdy(30,:) amount of pesticide lost through reactions in reach on day (mg pst)*
- rchdy(31,:) amount of pesticide lost through volatilization from reach on day (mg pst)*
- rchdy(32,:) amount of pesticide settling out of reach to bed sediment on day (mg pst)*
- rchdy(33,:) amount of pesticide resuspended from bed sediment to reach on day (mg pst)*
- rchdy(34,:) amount of pesticide diffusing from reach to bed sediment on day (mg pst)*
- rchdy(35,:) amount of pesticide in sediment layer lost through reactions on day (mg pst)*
- rchdy(36,:) amount of pesticide in sediment layer lost through burial on day (mg pst)*
- rchdy(37,:) amount of pesticide stored in river bed sediments (mg pst)*
- rchdy(38,:) persistent bacteria transported out of reach on day (kg bact)*
- rchdy(39,:) less persistent bacteria transported out of reach on day (kg bact)*
- rchdy(40,:) amount of conservative metal #1 transported out of reach on day (kg)*
- rchdy(41,:) amount of conservative metal #2 transported out of reach on day (kg)*
- rchdy(42,:) amount of conservative metal #3 transported out of reach on day (kg)*
- rchdy(43,:) total N (org N + no3 + no2 + nh4 outs) (kg)*
- rchdy(44,:) total P (org P + sol p outs) (kg)*
- real \*8, dimension(:,:), allocatable **hruyro**
  - HRU annual output array (varies) hruyro(1,:) precipitation in HRU during year (mm H2O)*
  - hruyro(2,:) amount of precipitation falling as freezing rain/snow in HRU during year (mm H2O)*
  - hruyro(3,:) amount of snow melt in HRU during year (mm H2O)*
  - hruyro(4,:) amount of surface runoff to main channel from HRU during year (ignores impact of transmission losses) (mm H2O)*
  - hruyro(5,:) amount of lateral flow contribution to main channel from HRU during year (mm H2O)*
  - hruyro(6,:) amount of groundwater flow contribution to main channel from HRU during year (mm H2O)*
  - hruyro(7,:) amount of water moving from shallow aquifer to plants or soil profile in HRU during year (mm H2O)*
  - hruyro(8,:) amount of water recharging deep aquifer in HRU during year (mm H2O)*
  - hruyro(9,:) total amount of water entering both aquifers from HRU during year (mm H2O)*
  - hruyro(10,:) water yield (total amount of water entering main channel) from HRU during year (mm H2O)*
  - hruyro(11,:) amount of water percolating out of the soil profile and into the vadose zone in HRU during year (mm H2O)*
  - hruyro(12,:) actual evapotranspiration in HRU during year (mm H2O)*
  - hruyro(13,:) amount of transmission losses from tributary channels in HRU for year (mm H2O)*
  - hruyro(14,:) sediment yield from HRU for year (metric tons/ha)*
  - hruyro(15,:) actual amount of transpiration that occurs during year in HRU (mm H2O)*
  - hruyro(16,:) actual amount of evaporation (from soil) that occurs during year in HRU (mm H2O)*
  - hruyro(17,:) amount of nitrogen applied in continuous fertilizer operation during year in HRU (kg N/ha)*
  - hruyro(18,:) amount of phosphorus applied in continuous fertilizer operation during year in HRU (kg P/ha)*
  - hruyro(23,:) amount of water removed from shallow aquifer in HRU for irrigation during year (mm H2O)*
  - hruyro(24,:) amount of water removed from deep aquifer in HRU for irrigation during year (mm H2O)*
  - hruyro(25,:) potential evapotranspiration in HRU during year (mm H2O)*
  - hruyro(26,:) annual amount of N (organic & mineral) applied in HRU during grazing (kg N/ha)*
  - hruyro(27,:) annual amount of P (organic & mineral) applied in HRU during grazing (kg P/ha)*
  - hruyro(28,:) annual amount of N (organic & mineral) auto-applied in HRU (kg N/ha)*
  - hruyro(29,:) annual amount of P (organic & mineral) auto-applied in HRU (kg P/ha)*
  - hruyro(31,:) water stress days in HRU during year (stress days)*
  - hruyro(32,:) temperature stress days in HRU during year (stress days)*



- hruyro(33,:) nitrogen stress days in HRU during year (stress days)*
  - hruyro(34,:) phosphorus stress days in HRU during year (stress days)*
  - hruyro(35,:) organic nitrogen in surface runoff in HRU during year (kg N/ha)*
  - hruyro(36,:) organic phosphorus in surface runoff in HRU during year (kg P/ha)*
  - hruyro(37,:) nitrate in surface runoff in HRU during year (kg N/ha)*
  - hruyro(38,:) nitrate in lateral flow in HRU during year (kg N/ha)*
  - hruyro(39,:) soluble phosphorus in surface runoff in HRU during year (kg P/ha)*
  - hruyro(40,:) amount of nitrogen removed from soil by plant uptake in HRU during year (kg N/ha)*
  - hruyro(41,:) nitrate percolating past bottom of soil profile in HRU during year (kg N/ha)*
  - hruyro(42,:) amount of phosphorus removed from soil by plant uptake in HRU during year (kg P/ha)*
  - hruyro(43,:) amount of phosphorus moving from labile mineral to active mineral pool in HRU during year (kg P/ha)*
  - hruyro(44,:) amount of phosphorus moving from active mineral to stable mineral pool in HRU during year (kg P/ha)*
  - hruyro(45,:) amount of nitrogen applied to HRU in fertilizer and grazing operations during year (kg N/ha)*
  - hruyro(46,:) amount of phosphorus applied to HRU in fertilizer and grazing operations during year (kg P/ha)*
  - hruyro(47,:) amount of nitrogen added to soil by fixation in HRU during year (kg N/ha)*
  - hruyro(48,:) amount of nitrogen lost by denitrification in HRU during year (kg N/ha)*
  - hruyro(49,:) amount of nitrogen moving from active organic to nitrate pool in HRU during year (kg N/ha)*
  - hruyro(50,:) amount of nitrogen moving from active organic to stable organic pool in HRU during year (kg N/ha)*
  - hruyro(51,:) amount of phosphorus moving from organic to labile mineral pool in HRU during year (kg P/ha)*
  - hruyro(52,:) amount of nitrogen moving from fresh organic to nitrate and active organic pools in HRU during year (kg N/ha)*
  - hruyro(53,:) amount of phosphorus moving from fresh organic to the labile mineral and organic pools in HRU during year (kg P/ha)*
  - hruyro(54,:) amount of nitrogen added to soil in rain during year (kg N/ha)*
  - hruyro(61,:) daily soil loss predicted with USLE equation (metric tons/ha)*
  - hruyro(63,:) less persistent bacteria transported to main channel from HRU during year (# bacteria/ha)*
  - hruyro(64,:) persistent bacteria transported to main channel from HRU during year (# bacteria/ha)*
  - hruyro(65,:) nitrate loading from groundwater in HRU to main channel during year (kg N/ha)*
  - hruyro(66,:) soluble P loading from groundwater in HRU to main channel during year (kg P/ha)*
  - hruyro(67,:) loading of mineral P attached to sediment in HRU to main channel during year (kg P/ha)*
- real \*8, dimension(:,:), allocatable **rchaao**
    - reach average annual output array (varies)*
    - rchaao(1,:) flow into reach during simulation ( $m^3/s$ )*
    - rchaao(2,:) flow out of reach during simulation ( $m^3/s$ )*
    - rchaao(3,:) sediment transported into reach during simulation (metric tons)*
    - rchaao(4,:) sediment transported out of reach during simulation (metric tons)*
    - rchaao(5,:) sediment concentration in outflow during simulation (mg/L)*
    - rchaao(6,:) organic N transported into reach during simulation (kg N)*
    - rchaao(7,:) organic N transported out of reach during simulation (kg N)*
    - rchaao(8,:) organic P transported into reach during simulation (kg P)*
    - rchaao(9,:) organic P transported out of reach during simulation (kg P)*
    - rchaao(10,:) evaporation from reach during simulation ( $m^3/s$ )*
    - rchaao(11,:) transmission losses from reach during simulation ( $m^3/s$ )*
    - rchaao(12,:) conservative metal #1 transported out of reach during simulation (kg)*
    - rchaao(13,:) conservative metal #2 transported out of reach during simulation (kg)*
    - rchaao(14,:) conservative metal #3 transported out of reach during simulation (kg)*
    - rchaao(15,:) nitrate transported into reach during simulation (kg N)*
    - rchaao(16,:) nitrate transported out of reach during simulation (kg N)*
    - rchaao(17,:) soluble P transported into reach during simulation (kg P)*
    - rchaao(18,:) soluble P transported out of reach during simulation (kg P)*
    - rchaao(19,:) soluble pesticide transported into reach during simulation*
    - rchaao(20,:) soluble pesticide transported out of reach during simulation*
    - rchaao(21,:) sorbed pesticide transported into reach during simulation*
    - rchaao(22,:) sorbed pesticide transported out of reach during simulation*
    - rchaao(23,:) amount of pesticide lost through reactions in reach during simulation*
    - rchaao(24,:) amount of pesticide lost through volatilization from reach during simulation*
    - rchaao(25,:) amount of pesticide settling out of reach to bed sediment during simulation*
    - rchaao(26,:) amount of pesticide resuspended from bed sediment to reach during simulation*
    - rchaao(27,:) amount of pesticide diffusing from reach to bed sediment during simulation*
    - rchaao(28,:) amount of pesticide in sediment layer lost through reactions during simulation*
    - rchaao(29,:) amount of pesticide in sediment layer lost through burial during simulation*
    - rchaao(30,:) chlorophyll-a transported into reach during simulation (kg chl a)*
    - rchaao(31,:) chlorophyll-a transported out of reach during simulation (kg chl a)*

- rchaao(32,:) ammonia transported into reach during simulation (kg N)*
- rchaao(33,:) ammonia transported out of reach during simulation (kg N)*
- rchaao(34,:) nitrite transported into reach during simulation (kg N)*
- rchaao(35,:) nitrite transported out of reach during simulation (kg N)*
- rchaao(36,:) CBOD transported into reach during simulation (kg O2)*
- rchaao(37,:) CBOD transported out of reach during simulation (kg O2)*
- rchaao(38,:) dissolved oxygen transported into reach during simulation (kg O2)*
- rchaao(39,:) dissolved oxygen transported out of reach during simulation (kg O2)*
- rchaao(40,:) persistent bacteria transported out of reach during simulation (kg bact)*
- rchaao(41,:) less persistent bacteria transported out of reach during simulation (kg bact)*
- rchaao(43,:) Total N (org N + no3 + no2 + nh4 outs) (kg)*
- rchaao(44,:) Total P (org P + sol p outs) (kg)*
- real \*8, dimension(:,:), allocatable [submono](#)
  - subbasin monthly output array (varies)*
  - submono(1,:) precipitation in subbasin for month (mm H2O)*
  - submono(2,:) snow melt in subbasin for month (mm H2O)*
  - submono(3,:) surface runoff loading in subbasin for month (mm H2O)*
  - submono(4,:) water yield from subbasin for month (mm H2O)*
  - submono(5,:) potential evapotranspiration in subbasin for month (mm H2O)*
  - submono(6,:) actual evapotranspiration in subbasin for month (mm H2O)*
  - submono(7,:) sediment yield from subbasin for month (metric tons/ha)*
  - submono(8,:) organic N loading from subbasin for month (kg N/ha)*
  - submono(9,:) organic P loading from subbasin for month (kg P/ha)*
  - submono(10,:) NO3 loading from surface runoff in subbasin for month (kg N/ha)*
  - submono(11,:) soluble P loading from subbasin for month (kg P/ha)*
  - submono(12,:) groundwater loading from subbasin for month (mm H2O)*
  - submono(13,:) percolation out of soil profile in subbasin for month (mm H2O)*
  - submono(14,:) loading to reach of mineral P attached to sediment from subbasin for month (kg P/ha)*
- real \*8, dimension(:,:), allocatable [subyro](#)
  - subbasin annual output array (varies)*
  - subyro(1,:) precipitation in subbasin for year (mm H2O)*
  - subyro(2,:) snow melt in subbasin for year (mm H2O)*
  - subyro(3,:) surface runoff loading in subbasin for year (mm H2O)*
  - subyro(4,:) water yield from subbasin for year (mm H2O)*
  - subyro(5,:) potential evapotranspiration in subbasin for year (mm H2O)*
  - subyro(6,:) actual evapotranspiration in subbasin for year (mm H2O)*
  - subyro(7,:) sediment yield from subbasin for year (metric tons/ha)*
  - subyro(8,:) organic N loading from subbasin for year (kg N/ha)*
  - subyro(9,:) organic P loading from subbasin for year (kg P/ha)*
  - subyro(10,:) NO3 loading from surface runoff in subbasin for year (kg N/ha)*
  - subyro(11,:) soluble P loading from subbasin for year (kg P/ha)*
  - subyro(12,:) groundwater loading from subbasin for year (mm H2O)*
  - subyro(13,:) percolation out of soil profile in subbasin for year (mm H2O)*
  - subyro(14,:) loading to reach of mineral P attached to sediment from subbasin for year (kg P/ha)*
- real \*8, dimension(:,:), allocatable [hruaao](#)
  - HRU average annual output array (varies)*
  - hruaao(1,:) precipitation in HRU during simulation (mm H2O)*
  - hruaao(2,:) amount of precipitation falling as freezing rain/snow in HRU during simulation (mm H2O)*
  - hruaao(3,:) amount of snow melt in HRU during simulation (mm H2O)*
  - hruaao(4,:) amount of surface runoff to main channel from HRU during simulation (ignores impact of transmission losses) (mm H2O)*
  - hruaao(5,:) amount of lateral flow contribution to main channel from HRU during simulation (mm H2O)*
  - hruaao(6,:) amount of groundwater flow contribution to main channel from HRU during simulation (mm H2O)*
  - hruaao(7,:) amount of water moving from shallow aquifer to plants or soil profile in HRU during simulation (mm H2O)*
  - hruaao(8,:) amount of water recharging deep aquifer in HRU during simulation (mm H2O)*
  - hruaao(9,:) total amount of water entering both aquifers from HRU during simulation (mm H2O)*
  - hruaao(10,:) water yield (total amount of water entering main channel) from HRU during simulation (mm H2O)*
  - hruaao(11,:) amount of water percolating out of the soil profile and into the vadose zone in HRU during simulation (mm H2O)*
  - hruaao(12,:) actual evapotranspiration in HRU during simulation*
  - hruaao(13,:) amount of transmission losses from tributary channels in HRU for simulation (mm H2O)*
  - hruaao(14,:) sediment yield from HRU for simulation (metric tons/ha)*

- hruaao(15,:)* actual amount of transpiration that occurs during simulation in HRU (mm H<sub>2</sub>O)
  - hruaao(16,:)* actual amount of evaporation (from soil) that occurs during simulation in HRU (mm H<sub>2</sub>O)
  - hruaao(17,:)* amount of nitrogen applied in continuous fertilizer operation in HRU for simulation (kg N/ha)
  - hruaao(18,:)* amount of phosphorus applied in continuous fertilizer operation in HRU for simulation (kg P/ha)
  - hruaao(22,:)* amount of irrigation water applied to HRU during simulation (mm H<sub>2</sub>O)
  - hruaao(23,:)* amount of water removed from shallow aquifer in HRU for irrigation during simulation (mm H<sub>2</sub>O)
  - hruaao(24,:)* amount of water removed from deep aquifer in HRU for irrigation during simulation (mm H<sub>2</sub>O)
  - hruaao(25,:)* potential evapotranspiration in HRU during simulation (mm H<sub>2</sub>O)
  - hruaao(26,:)* annual amount of N (organic & mineral) applied in HRU during grazing (kg N/ha)
  - hruaao(27,:)* annual amount of P (organic & mineral) applied in HRU during grazing (kg P/ha)
  - hruaao(28,:)* average annual amount of N (organic & mineral) auto-applied in HRU (kg N/ha)
  - hruaao(29,:)* average annual amount of P (organic & mineral) auto-applied in HRU (kg P/ha)
  - hruaao(31,:)* water stress days in HRU during simulation (stress days)
  - hruaao(32,:)* temperature stress days in HRU during simulation (stress days)
  - hruaao(33,:)* nitrogen stress days in HRU during simulation (stress days)
  - hruaao(34,:)* phosphorus stress days in HRU during simulation (stress days)
  - hruaao(35,:)* organic nitrogen in surface runoff in HRU during simulation (kg N/ha)
  - hruaao(36,:)* organic phosphorus in surface runoff in HRU during simulation (kg P/ha)
  - hruaao(37,:)* nitrate in surface runoff in HRU during simulation (kg N/ha)
  - hruaao(38,:)* nitrate in lateral flow in HRU during simulation (kg N/ha)
  - hruaao(39,:)* soluble phosphorus in surface runoff in HRU during simulation (kg P/ha)
  - hruaao(40,:)* amount of nitrogen removed from soil by plant uptake in HRU during simulation (kg N/ha)
  - hruaao(41,:)* nitrate percolating past bottom of soil profile in HRU during simulation (kg N/ha)
  - hruaao(42,:)* amount of phosphorus removed from soil by plant uptake in HRU during simulation (kg P/ha)
  - hruaao(43,:)* amount of phosphorus moving from labile mineral to active mineral pool in HRU during simulation (kg P/ha)
  - hruaao(44,:)* amount of phosphorus moving from active mineral to stable mineral pool in HRU during simulation (kg P/ha)
  - hruaao(45,:)* amount of nitrogen applied to HRU in fertilizer and grazing operations during simulation (kg N/ha)
  - hruaao(46,:)* amount of phosphorus applied to HRU in fertilizer and grazing operations during simulation (kg P/ha)
  - hruaao(47,:)* amount of nitrogen added to soil by fixation in HRU during simulation (kg N/ha)
  - hruaao(48,:)* amount of nitrogen lost by denitrification in HRU during simulation (kg N/ha)
  - hruaao(49,:)* amount of nitrogen moving from active organic to nitrate pool in HRU during simulation (kg N/ha)
  - hruaao(50,:)* amount of nitrogen moving from active organic to stable organic pool in HRU during simulation (kg N/ha)
  - hruaao(51,:)* amount of phosphorus moving from organic to labile mineral pool in HRU during simulation (kg P/ha)
  - hruaao(52,:)* amount of nitrogen moving from fresh organic to nitrate and active organic pools in HRU during simulation (kg N/ha)
  - hruaao(53,:)* amount of phosphorus moving from fresh organic to the labile mineral and organic pools in HRU during simulation (kg P/ha)
  - hruaao(54,:)* amount of nitrogen added to soil in rain during simulation (kg N/ha)
  - hruaao(61,:)* daily soil loss predicted with USLE equation (metric tons/ha)
  - hruaao(63,:)* less persistent bacteria transported to main channel from HRU during simulation (# bacteria/ha)
  - hruaao(64,:)* persistent bacteria transported to main channel from HRU during simulation (# bacteria/ha)
  - hruaao(65,:)* nitrate loading from groundwater in HRU to main channel during simulation (kg N/ha)
  - hruaao(66,:)* soluble P loading from groundwater in HRU to main channel during simulation (kg P/ha)
  - hruaao(67,:)* loading of mineral P attached to sediment in HRU to main channel during simulation (kg P/ha)
- real \*8, dimension(:,), allocatable **subaao**
    - subbasin* average annual output array (varies)
    - subaao(1,:)* precipitation in subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(2,:)* snow melt in subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(3,:)* surface runoff loading in subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(4,:)* water yield from subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(5,:)* potential evapotranspiration in subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(6,:)* actual evapotranspiration in subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(7,:)* sediment yield from subbasin for simulation (metric tons/ha)
    - subaao(8,:)* organic N loading from subbasin for simulation (kg N/ha)
    - subaao(9,:)* organic P loading from subbasin for simulation (kg P/ha)
    - subaao(10,:)* NO<sub>3</sub> loading from surface runoff in subbasin for simulation (kg N/ha)
    - subaao(11,:)* soluble P loading from subbasin for simulation (kg P/ha)
    - subaao(12,:)* groundwater loading from subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(13,:)* percolation out of soil profile in subbasin for simulation (mm H<sub>2</sub>O)
    - subaao(14,:)* loading to reach of mineral P attached to sediment from subbasin for simulation (kg P/ha)
    - subaao(18,i)* groundwater?

- real \*8, dimension(:,:), allocatable **resoutm**

*reservoir monthly output array (varies)*

*resoutm(1,:) flow into reservoir during month ( $m^3/s$ )*  
*resoutm(2,:) flow out of reservoir during month ( $m^3/s$ )*  
*resoutm(3,:) sediment entering reservoir during month (metric tons)*  
*resoutm(4,:) sediment leaving reservoir during month (metric tons)*  
*resoutm(5,:) sediment concentration in reservoir during month (mg/L)*  
*resoutm(6,:) pesticide entering reservoir during month (mg pst)*  
*resoutm(7,:) pesticide lost from reservoir through reactions during month (mg pst)*  
*resoutm(8,:) pesticide lost from reservoir through volatilization during month (mg pst)*  
*resoutm(9,:) pesticide moving from water to sediment through settling during month (mg pst)*  
*resoutm(10,:) pesticide moving from sediment to water through resuspension during month (mg pst)*  
*resoutm(11,:) pesticide moving from water to sediment through diffusion during month (mg pst)*  
*resoutm(12,:) pesticide lost from reservoir sediment layer through reactions during month (mg pst)*  
*resoutm(13,:) pesticide lost from reservoir sediment layer through burial during month (mg pst)*  
*resoutm(14,:) pesticide transported out of reservoir during month (mg pst)*  
*resoutm(15,:) pesticide concentration in reservoir water during month (mg pst/ $m^3$ )*  
*resoutm(16,:) pesticide concentration in reservoir sediment layer during month (mg pst/ $m^3$ )*  
*resoutm(17,:) evaporation from reservoir during month ( $m^3 H_2O$ )*  
*resoutm(18,:) seepage from reservoir during month ( $m^3 H_2O$ )*  
*resoutm(19,:) precipitation on reservoir during month ( $m^3 H_2O$ )*  
*resoutm(20,:) water flowing into reservoir during month ( $m^3 H_2O$ )*  
*resoutm(21,:) water flowing out of reservoir during month ( $m^3 H_2O$ )*  
*resoutm(22,:) organic N entering reservoir during month (kg N)*  
*resoutm(23,:) organic N leaving reservoir during month (kg N)*  
*resoutm(24,:) organic P entering reservoir during month (kg P)*  
*resoutm(25,:) organic P leaving reservoir during month (kg P)*  
*resoutm(26,:) nitrate entering reservoir during month (kg N)*  
*resoutm(27,:) nitrate leaving reservoir during month (kg N)*  
*resoutm(28,:) nitrite entering reservoir during month (kg N)*  
*resoutm(29,:) nitrite leaving reservoir during month (kg N)*  
*resoutm(30,:) ammonia entering reservoir during month (kg N)*  
*resoutm(31,:) ammonia leaving reservoir during month (kg N)*  
*resoutm(32,:) mineral P entering reservoir during month (kg P)*  
*resoutm(33,:) mineral P leaving reservoir during month (kg P)*  
*resoutm(34,:) chlorophyll-a entering reservoir during month (kg chla)*  
*resoutm(35,:) chlorophyll-a leaving reservoir during month (kg chla)*  
*resoutm(36,:) organic P concentration in reservoir water during month (mg P/L)*  
*resoutm(37,:) mineral P concentration in reservoir water during month (mg P/L)*  
*resoutm(38,:) organic N concentration in reservoir water during month (mg N/L)*  
*resoutm(39,:) nitrate concentration in reservoir water during month (mg N/L)*  
*resoutm(40,:) nitrite concentration in reservoir water during month (mg N/L)*  
*resoutm(41,:) ammonia concentration in reservoir water during month (mg N/L)*

- real \*8, dimension(:,:), allocatable **resouty**

*reservoir annual output array (varies)*

*resouty(1,:) flow into reservoir during year ( $m^3/s$ )*  
*resouty(2,:) flow out of reservoir during year ( $m^3/s$ )*  
*resouty(3,:) sediment entering reservoir during year (metric tons)*  
*resouty(4,:) sediment leaving reservoir during year (metric tons)*  
*resouty(5,:) sediment concentration in reservoir during year (mg/L)*  
*resouty(6,:) pesticide entering reservoir during year (mg pst)*  
*resouty(7,:) pesticide lost from reservoir through reactions during year (mg pst)*  
*resouty(8,:) pesticide lost from reservoir through volatilization during year (mg pst)*  
*resouty(9,:) pesticide moving from water to sediment through settling during year (mg pst)*  
*resouty(10,:) pesticide moving from sediment to water through resuspension during year (mg pst)*  
*resouty(11,:) pesticide moving from water to sediment through diffusion during year (mg pst)*  
*resouty(12,:) pesticide lost from reservoir sediment layer through reactions during year (mg pst)*  
*resouty(13,:) pesticide lost from reservoir sediment layer through burial during year (mg pst)*  
*resouty(14,:) pesticide transported out of reservoir during year (mg pst)*  
*resouty(15,:) pesticide concentration in reservoir water during year (mg pst/ $m^3$ )*  
*resouty(16,:) pesticide concentration in reservoir sediment layer during year (mg pst/ $m^3$ )*  
*resouty(17,:) evaporation from reservoir during year ( $m^3 H_2O$ )*  
*resouty(18,:) seepage from reservoir during year ( $m^3 H_2O$ )*



- resouty(19,:) precipitation on reservoir during year (m<sup>3</sup> H<sub>2</sub>O)*
- resouty(22,:) organic N entering reservoir during year (kg N)*
- resouty(23,:) organic N leaving reservoir during year (kg N)*
- resouty(24,:) organic P entering reservoir during year (kg P)*
- resouty(25,:) organic P leaving reservoir during year (kg P)*
- resouty(26,:) nitrate entering reservoir during year (kg N)*
- resouty(27,:) nitrate leaving reservoir during year (kg N)*
- resouty(28,:) nitrite entering reservoir during year (kg N)*
- resouty(29,:) nitrite leaving reservoir during year (kg N)*
- resouty(30,:) ammonia entering reservoir during year (kg N)*
- resouty(31,:) ammonia leaving reservoir during year (kg N)*
- resouty(32,:) mineral P entering reservoir during year (kg P)*
- resouty(33,:) mineral P leaving reservoir during year (kg P)*
- resouty(34,:) chlorophyll-a entering reservoir during year (kg chla)*
- resouty(35,:) chlorophyll-a leaving reservoir during year (kg chla)*
- resouty(36,:) organic P concentration in reservoir water during year (mg P/L)*
- resouty(37,:) mineral P concentration in reservoir water during year (mg P/L)*
- resouty(38,:) organic N concentration in reservoir water during year (mg N/L)*
- resouty(39,:) nitrate concentration in reservoir water during year (mg N/L)*
- resouty(40,:) nitrite concentration in reservoir water during year (mg N/L)*
- resouty(41,:) ammonia concentration in reservoir water during year (mg N/L)*
- real \*8, dimension(:,:), allocatable [resouta](#)
  - reservoir average annual output array (varies)*
  - resouta(3,:) sediment entering reservoir during simulation (metric tons)*
  - resouta(4,:) sediment leaving reservoir during simulation (metric tons)*
  - resouta(17,:) evaporation from reservoir during simulation (m<sup>3</sup> H<sub>2</sub>O)*
  - resouta(18,:) seepage from reservoir during simulation (m<sup>3</sup> H<sub>2</sub>O)*
  - resouta(19,:) precipitation on reservoir during simulation (m<sup>3</sup> H<sub>2</sub>O)*
  - resouta(20,:) water entering reservoir during simulation (m<sup>3</sup> H<sub>2</sub>O)*
  - resouta(21,:) water leaving reservoir during simulation (m<sup>3</sup> H<sub>2</sub>O)*
- real \*8, dimension(12, 8) [wshd\\_aamon](#)
  - array of watershed monthly average values (varies)*
  - wshd\_aamon(:,1) average annual precipitation in watershed falling during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,2) average annual freezing rain in watershed falling during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,3) average annual surface runoff in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,4) average annual lateral flow in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,5) average annual water yield in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,6) average annual actual evapotranspiration in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,7) average annual sediment yield in watershed during month (metric tons)*
  - wshd\_aamon(:,8) average annual potential evapotranspiration in watershed during month (mm H<sub>2</sub>O)*
- real \*8, dimension(:,:), allocatable [wtrmon](#)
  - HRU monthly output data array for impoundments (varies)*
  - wtrmon(1,:) evaporation from ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(2,:) seepage from ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(3,:) precipitation on ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(4,:) amount of water entering ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(5,:) sediment entering ponds in HRU for month (metric tons/ha)*
  - wtrmon(6,:) amount of water leaving ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(7,:) sediment leaving ponds in HRU for month (metric tons/ha)*
  - wtrmon(8,:) precipitation on wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(9,:) volume of water entering wetlands from HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(10,:) sediment loading to wetlands from HRU for month (metric tons/ha)*
  - wtrmon(11,:) evaporation from wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(12,:) seepage from wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(13,:) volume of water leaving wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(14,:) sediment loading from wetlands in HRU to main channel during month (metric tons/ha)*
  - wtrmon(15,:) precipitation on potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(16,:) evaporation from potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(17,:) seepage from potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(18,:) water leaving potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(19,:) water entering potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(20,:) sediment entering potholes in HRU for month (metric tons/ha)*

- wtrmon(21,:) sediment leaving potholes in HRU for month (metric tons/ha)*

  - real \*8, dimension(:,:), allocatable [wtryr](#)

HRU impoundment annual output array (varies)

*wtryr(1,:) evaporation from ponds in HRU for year (mm H2O)*

*wtryr(2,:) seepage from ponds in HRU for year (mm H2O)*

*wtryr(3,:) precipitation on ponds in HRU for year (mm H2O)*

*wtryr(4,:) amount of water entering ponds in HRU for year (mm H2O)*

*wtryr(5,:) sediment entering ponds in HRU for year (metric tons/ha)*

*wtryr(6,:) amount of water leaving ponds in HRU for year (mm H2O)*

*wtryr(7,:) sediment leaving ponds in HRU for year (metric tons/ha)*

*wtryr(8,:) precipitation on wetlands in HRU for year (mm H2O)*

*wtryr(9,:) volume of water entering wetlands from HRU for year (mm H2O)*

*wtryr(10,:) sediment loading to wetlands for year from HRU (metric tons/ha)*

*wtryr(11,:) evaporation from wetlands in HRU for year (mm H2O)*

*wtryr(12,:) seepage from wetlands in HRU for year (mm H2O)*

*wtryr(13,:) volume of water leaving wetlands in HRU for year (mm H2O)*

*wtryr(14,:) sediment loading from wetlands in HRU to main channel during year (metric tons/ha)*

*wtryr(15,:) precipitation on potholes in HRU during year (mm H2O)*

*wtryr(16,:) evaporation from potholes in HRU during year (mm H2O)*

*wtryr(17,:) seepage from potholes in HRU during year (mm H2O)*

*wtryr(18,:) water leaving potholes in HRU during year (mm H2O)*

*wtryr(19,:) water entering potholes in HRU during year (mm H2O)*

*wtryr(20,:) sediment entering potholes in HRU during year (metric tons/ha)*

*wtryr(21,:) sediment leaving potholes in HRU during year (metric tons/ha)*
- real \*8, dimension(:,:), allocatable [wtraa](#)

HRU impoundment average annual output array (varies)

*wtraa(1,:) evaporation from ponds in HRU during simulation (mm H2O)*

*wtraa(2,:) seepage from ponds in HRU during simulation (mm H2O)*

*wtraa(3,:) precipitation on ponds in HRU during simulation (mm H2O)*

*wtraa(4,:) amount of water entering ponds in HRU during simulation (mm H2O)*

*wtraa(5,:) sediment entering ponds in HRU during simulation (metric tons/ha)*

*wtraa(6,:) amount of water leaving ponds in HRU during simulation (mm H2O)*

*wtraa(7,:) sediment leaving ponds in HRU during simulation (metric tons/ha)*

*wtraa(8,:) precipitation on wetlands in HRU during simulation (mm H2O)*

*wtraa(9,:) volume of water entering wetlands from HRU during simulation (mm H2O)*

*wtraa(10,:) sediment loading to wetlands during simulation from HRU (metric tons/ha)*

*wtraa(11,:) evaporation from wetlands in HRU during simulation (mm H2O)*

*wtraa(12,:) seepage from wetlands in HRU during simulation (mm H2O)*

*wtraa(13,:) volume of water leaving wetlands in HRU during simulation (mm H2O)*

*wtraa(14,:) sediment loading from wetlands in HRU to main channel during simulation (metric tons/ha)*

*wtraa(15,:) precipitation on potholes in HRU during simulation (mm H2O)*

*wtraa(16,:) evaporation from potholes in HRU during simulation (mm H2O)*

*wtraa(17,:) seepage from potholes in HRU during simulation (mm H2O)*

*wtraa(18,:) water leaving potholes in HRU during simulation (mm H2O)*

*wtraa(19,:) water entering potholes in HRU during simulation (mm H2O)*

*wtraa(20,:) sediment entering potholes in HRU during simulation (metric tons/ha)*

*wtraa(21,:) sediment leaving potholes in HRU during simulation (metric tons/ha)*
- real \*8, dimension(:,:), allocatable [sub\\_smfmfx](#)

*max melt rate for snow during year (June 21) for subbasin(:) where deg C refers to the air temperature. SUB\_SMFMX and SMFMN allow the rate of snow melt to vary through the year. These parameters are accounting for the impact of soil temperature on snow melt (range: -5.0/5.0) (mm/deg C/day)*
- real \*8, dimension(:,:), allocatable [sub\\_smfmn](#)

*min melt rate for snow during year (Dec 21) for subbasin(:) (range: -5.0/5.0) where deg C refers to the air temperature (mm/deg C/day)*
- real \*8, dimension(:,:), allocatable [hrupstd](#)

HRU daily pesticide output array (varies)

*hrupstd(1,:) amount of pesticide type in surface runoff contribution to stream from HRU on day (in solution) (mg pst)*

*hrupstd(2,:) amount of pesticide type in surface runoff contribution to stream from HRU on day (sorbed to sediment) (mg pst)*

*hrupstd(3,:) total pesticide loading to stream in surface runoff from HRU (mg pst/ha)*

*hrupstd(4,:) amount of pesticide type in lateral flow contribution to stream from HRU on day (in solution) (mg pst)*

- real \*8, dimension(:,:), allocatable [hrupstm](#)  
*hrupstm(:, :, :)* HRU monthly pesticide output array (varies)  
*hrupstm(1, :, :)* amount of pesticide type in surface runoff contribution to stream from HRU during month (in solution) (mg pst)  
*hrupstm(2, :, :)* amount of pesticide type in surface runoff contribution to stream from HRU during month (sorbed to sediment) (mg pst)  
*hrupstm(3, :, :)* total pesticide loading to stream in surface runoff from HRU during month (mg pst)
- real \*8, dimension(:,:), allocatable [hrupsta](#)  
*hrupsta* HRU average annual pesticide output array (varies)
- real \*8, dimension(:,:), allocatable [hrupsty](#)  
*hrupsty(:, :, :)* HRU annual pesticide output array (varies)  
*hrupsty(1, :, :)* amount of pesticide type in surface runoff contribution to stream from HRU during year (in solution) (mg pst)  
*hrupsty(2, :, :)* amount of pesticide type in surface runoff contribution to stream from HRU during year (sorbed to sediment) (mg pst)
- integer, dimension(:), allocatable [ifirstt](#)  
*ifirstt* temperature data search code (none)  
0 first day of temperature data located in file  
1 first day of temperature data not located in file
- integer, dimension(:), allocatable [ifirstpcp](#)
- integer, dimension(:), allocatable [elevp](#)  
*elevp* elevation of precipitation gage station (m)
- integer, dimension(:), allocatable [elevt](#)  
*elevt* elevation of temperature gage station (m)
- real \*8, dimension(:,:), allocatable [ftmpmn](#)  
*ftmpmn* avg monthly minimum air temperature (deg C)
- real \*8, dimension(:,:), allocatable [ftmpmx](#)  
*ftmpmx* avg monthly maximum air temperature (deg C)
- real \*8, dimension(:,:), allocatable [ftmpstdmn](#)  
*ftmpstdmn* standard deviation for avg monthly minimum air temperature (deg C)
- real \*8, dimension(:,:), allocatable [ftmpstdmx](#)  
*ftmpstdmx* standard deviation for avg monthly maximum air temperature (deg C)
- real \*8, dimension(:,:), allocatable [fpcp\\_stat](#)  
*fpcp\_stat(:, 1, :)* average amount of precipitation falling in one day for the month (mm/day)  
*fpcp\_stat(:, 2, :)* standard deviation for the average daily precipitation (mm/day)  
*fpcp\_stat(:, 3, :)* skew coefficient for the average daily precipitation (none)
- real \*8, dimension(:,:), allocatable [fpr\\_w](#)  
*fpr\_w(1, :, :)* probability of wet day after dry day in month (none)  
*fpr\_w(2, :, :)* probability of wet day after wet day in month (none)
- real \*8, dimension(:), allocatable [ch\\_d](#)  
*ch\_d* average depth of main channel (m)
- real \*8, dimension(:), allocatable [flwin](#)  
*flwin* flow into reach on current day ( $m^3 H_2O$ )
- real \*8, dimension(:), allocatable [flwout](#)  
*flwout* flow out of reach on current day ( $m^3 H_2O$ )
- real \*8, dimension(:), allocatable [bankst](#)  
*bankst* bank storage ( $m^3 H_2O$ )
- real \*8, dimension(:), allocatable [ch\\_wi](#)
- real \*8, dimension(:), allocatable [ch\\_onco](#)  
*ch\_onco* channel organic n concentration (ppm)
- real \*8, dimension(:), allocatable [ch\\_opco](#)  
*ch\_opco* channel organic p concentration (ppm)
- real \*8, dimension(:), allocatable [ch\\_orgn](#)
- real \*8, dimension(:), allocatable [ch\\_orgp](#)

- real \*8, dimension(:), allocatable [rch\\_dox](#)  
*dissolved oxygen concentration in reach (mg O2/L)*
- real \*8, dimension(:), allocatable [rch\\_bactp](#)  
*persistent bacteria in reach/outflow at end of day (# cfu/100ml)*
- real \*8, dimension(:), allocatable [alpha\\_bnk](#)  
*alpha factor for bank storage recession curve (days)*
- real \*8, dimension(:), allocatable [alpha\\_bnke](#)  
 $\exp(-\alpha_{bnk})$  (none)
- real \*8, dimension(:), allocatable [rchstor](#)  
*water stored in reach ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable [sedst](#)  
*amount of sediment stored in reach (metric tons)*
- real \*8, dimension(:), allocatable [algae](#)  
*algal biomass concentration in reach (mg alg/L)*
- real \*8, dimension(:), allocatable [disolvp](#)  
*dissolved phosphorus concentration in reach (mg P/L)*
- real \*8, dimension(:), allocatable [chlora](#)  
*chlorophyll-a concentration in reach (mg chl-a/L)*
- real \*8, dimension(:), allocatable [organicn](#)  
*organic nitrogen concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [organicp](#)  
*organic phosphorus concentration in reach (mg P/L)*
- real \*8, dimension(:), allocatable [ch\\_li](#)  
*initial length of main channel (km)*
- real \*8, dimension(:), allocatable [ch\\_si](#)  
*initial slope of main channel (m/m)*
- real \*8, dimension(:), allocatable [nitraten](#)  
*nitrate concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [nitriten](#)  
*nitrite concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [ch\\_bnk\\_san](#)
- real \*8, dimension(:), allocatable [ch\\_bnk\\_sil](#)
- real \*8, dimension(:), allocatable [ch\\_bnk\\_cla](#)
- real \*8, dimension(:), allocatable [ch\\_bnk\\_gra](#)
- real \*8, dimension(:), allocatable [ch\\_bed\\_san](#)
- real \*8, dimension(:), allocatable [ch\\_bed\\_sil](#)
- real \*8, dimension(:), allocatable [ch\\_bed\\_cla](#)
- real \*8, dimension(:), allocatable [ch\\_bed\\_gra](#)
- real \*8, dimension(:), allocatable [depfp](#)
- real \*8, dimension(:), allocatable [depsilfp](#)
- real \*8, dimension(:), allocatable [depclafp](#)
- real \*8, dimension(:), allocatable [depch](#)
- real \*8, dimension(:), allocatable [depsanch](#)
- real \*8, dimension(:), allocatable [depsilch](#)
- real \*8, dimension(:), allocatable [depclach](#)
- real \*8, dimension(:), allocatable [depsagch](#)
- real \*8, dimension(:), allocatable [deplagch](#)
- real \*8, dimension(:), allocatable [depgrach](#)
- real \*8, dimension(:), allocatable [grast](#)
- real \*8, dimension(:), allocatable [prf](#)  
*Reach peak rate adjustment factor for sediment routing in the channel. Allows impact of peak flow rate on sediment routing and channel reshaping to be taken into account (none)*



- real \*8, dimension(:), allocatable **depprch**
- real \*8, dimension(:), allocatable **depprpf**
- real \*8, dimension(:), allocatable **spcon**  
*linear parameter for calculating sediment reentrained in channel sediment routing*
- real \*8, dimension(:), allocatable **spexp**  
*exponent parameter for calculating sediment reentrained in channel sediment routing*
- real \*8, dimension(:), allocatable **sanst**
- real \*8, dimension(:), allocatable **silst**
- real \*8, dimension(:), allocatable **clast**
- real \*8, dimension(:), allocatable **sagst**
- real \*8, dimension(:), allocatable **lagst**
- real \*8, dimension(:), allocatable **pot\_san**
- real \*8, dimension(:), allocatable **pot\_sil**
- real \*8, dimension(:), allocatable **pot\_cla**
- real \*8, dimension(:), allocatable **pot\_sag**
- real \*8, dimension(:), allocatable **pot\_lag**
- real \*8, dimension(:), allocatable **sanyld**
- real \*8, dimension(:), allocatable **silyld**
- real \*8, dimension(:), allocatable **clayld**
- real \*8, dimension(:), allocatable **sagyld**
- real \*8, dimension(:), allocatable **lagyld**
- real \*8, dimension(:), allocatable **res\_san**
- real \*8, dimension(:), allocatable **res\_sil**
- real \*8, dimension(:), allocatable **res\_cla**
- real \*8, dimension(:), allocatable **res\_sag**
- real \*8, dimension(:), allocatable **res\_lag**
- real \*8, dimension(:), allocatable **res\_gra**
- real \*8, dimension(:), allocatable **pnd\_san**
- real \*8, dimension(:), allocatable **pnd\_sil**
- real \*8, dimension(:), allocatable **pnd\_cla**
- real \*8, dimension(:), allocatable **pnd\_sag**
- real \*8, dimension(:), allocatable **pnd\_lag**
- real \*8, dimension(:), allocatable **wet\_san**
- real \*8, dimension(:), allocatable **wet\_sil**
- real \*8, dimension(:), allocatable **wet\_cla**
- real \*8, dimension(:), allocatable **wet\_lag**
- real \*8, dimension(:), allocatable **wet\_sag**
- real \*8 **ressani**
- real \*8 **ressili**
- real \*8 **resclai**
- real \*8 **ressagi**
- real \*8 **reslagi**
- real \*8 **resgrai**
- real \*8 **pndsanin**
- real \*8 **pndsilin**
- real \*8 **pndclain**
- real \*8 **pndsagin**
- real \*8 **pndlagin**
- real \*8 **pndsano**
- real \*8 **pndsilo**
- real \*8 **pndclao**
- real \*8 **pndsago**
- real \*8 **pndlago**
- real \*8, dimension(:), allocatable **ch\_di**

- initial depth of main channel (m)*
- real \*8, dimension(:,:), allocatable [ch\\_l](#)
  - ch\_l(1,:) longest tributary channel length in subbasin (km)*
  - ch\_l(2,:) length of main channel (km)*
- real \*8, dimension(:), allocatable [ch\\_bnk\\_bd](#)
  - bulk density of channel bank sediment (1.1-1.9) (g/cc)*
- real \*8, dimension(:), allocatable [ch\\_bed\\_bd](#)
  - bulk density of channel bed sediment (1.1-1.9) (g/cc)*
- real \*8, dimension(:), allocatable [ch\\_bnk\\_kd](#)
  - erodibility of channel bank sediment by jet test (Peter Allen needs to give more info on this)*
- real \*8, dimension(:), allocatable [ch\\_bed\\_kd](#)
  - erodibility of channel bed sediment by jet test (Peter Allen needs to give more info on this)*
- real \*8, dimension(:), allocatable [ch\\_bnk\\_d50](#)
  - D50(median) particle size diameter of channel bank sediment (0.001 - 20)*
- real \*8, dimension(:), allocatable [ch\\_bed\\_d50](#)
  - D50(median) particle size diameter of channel bed sediment (micrometers) (0.001 - 20)*
- real \*8, dimension(:,:), allocatable [ch\\_cov](#)
  - ch\_cov(1,:) channel erodibility factor (0.0-1.0) (none)*
  - 0 non-erosive channel*
  - 1 no resistance to erosion*
  - ch\_cov(2,:) channel cover factor (0.0-1.0) (none)*
  - 0 channel is completely protected from erosion by cover*
  - 1 no vegetative cover on channel*
- real \*8, dimension(:), allocatable [tc\\_bed](#)
  - critical shear stress of channel bed (N/m2)*
- real \*8, dimension(:), allocatable [tc\\_bnk](#)
  - critical shear stress of channel bank (N/m2)*
- integer, dimension(:), allocatable [ch\\_eqn](#)
  - sediment routine methods (DAILY):*
  - 0 = original SWAT method*
  - 1 = Bagnold's*
  - 2 = Kodatie*
  - 3 = Molinas WU*
  - 4 = Yang*
- real \*8, dimension(:), allocatable [chpst\\_rea](#)
  - pesticide reaction coefficient in reach (1/day)*
- real \*8, dimension(:), allocatable [chpst\\_vol](#)
  - pesticide volatilization coefficient in reach (m/day)*
- real \*8, dimension(:), allocatable [chpst\\_conc](#)
  - initial pesticide concentration in reach (mg/(m<sup>3</sup>))*
- real \*8, dimension(:), allocatable [chpst\\_koc](#)
  - pesticide partition coefficient between water and sediment in reach (m<sup>3</sup>/g)*
- real \*8, dimension(:), allocatable [chpst\\_rsp](#)
  - resuspension velocity in reach for pesticide sorbed to sediment (m/day)*
- real \*8, dimension(:), allocatable [chpst\\_stl](#)
  - settling velocity in reach for pesticide sorbed to sediment (m/day)*
- real \*8, dimension(:), allocatable [ch\\_wdr](#)
  - channel width to depth ratio (m/m)*
- real \*8, dimension(:), allocatable [chpst\\_mix](#)
  - mixing velocity (diffusion/dispersion) for pesticide in reach (m/day)*
- real \*8, dimension(:), allocatable [sedpst\\_conc](#)
  - initil pesticide concentration in river bed sediment (mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable [sedpst\\_bry](#)

- pesticide burial velocity in river bed sediment (m/day)*
- real \*8, dimension(:), allocatable [sedpst\\_rea](#)  
*pesticide reaction coefficient in river bed sediment (1/day)*
- real \*8, dimension(:), allocatable [sedpst\\_act](#)  
*depth of active sediment layer in reach for pesticide (m)*
- real \*8, dimension(:), allocatable [rch\\_cbod](#)  
*carbonaceous biochemical oxygen demand in reach (mg O2/L)*
- real \*8, dimension(:), allocatable [rch\\_bactlp](#)  
*less persistent bacteria in reach/outflow at end of day (# cfu/100ml)*
- real \*8, dimension(:), allocatable [chside](#)  
*change in horizontal distance per unit vertical distance (0.0 - 5)*  
*0 = for vertical channel bank*  
*5 = for channel bank with gentl side slope*
- real \*8, dimension(:,:), allocatable [rs](#)  
*rs(1,:) local algal settling rate in reach at 20 deg C (m/day or m/hour)*  
*rs(2,:) benthos source rate for dissolved phosphorus in reach at 20 deg C ((mg disP-P)/(m<sup>2</sup>\*day) or (mg disP-P)/(m<sup>2</sup>\*hour))*  
*rs(3,:) benthos source rate for ammonia nitrogen in reach at 20 deg C ((mg NH4-N)/(m<sup>2</sup>\*day) or (mg NH4-N)/(m<sup>2</sup>\*hour))*  
*rs(4,:) rate coefficient for organic nitrogen settling in reach at 20 deg C (1/day or 1/hour)*  
*rs(5,:) organic phosphorus settling rate in reach at 20 deg C (1/day or 1/hour)*  
*rs(6,:) rate coefficient for settling of arbitrary non-conservative constituent in reach (1/day)*  
*rs(7,:) benthos source rate for arbitrary non-conservative constituent in reach ((mg ANC)/(m<sup>2</sup>\*day))*
- real \*8, dimension(:,:), allocatable [rk](#)  
*rk(1,:) CBOD deoxygenation rate coefficient in reach at 20 deg C (1/day or 1/hour)*  
*rk(2,:) reaeration rate in accordance with Fickian diffusion in reach at 20 deg C (1/day or 1/hour)*  
*rk(3,:) rate of loss of CBOD due to settling in reach at 20 deg C (1/day or 1/hour)*  
*rk(4,:) sediment oxygen demand rate in reach at 20 deg C (mg O2/(m<sup>2</sup>\*day) or mg O2/(m<sup>2</sup>\*hour))*  
*rk(5,:) coliform die-off rate in reach (1/day)*  
*rk(6,:) decay rate for arbitrary non-conservative constituent in reach (1/day)*
- real \*8, dimension(:,:), allocatable [bc](#)  
*bc(1,:) rate constant for biological oxidation of NH3 to NO2 in reach at 20 deg C (1/day or 1/hour)*  
*bc(2,:) rate constant for biological oxidation of NO2 to NO3 in reach at 20 deg C (1/day or 1/hour)*  
*bc(3,:) rate constant for hydrolysis of organic N to ammonia in reach at 20 deg C (1/day or 1/hour)*  
*bc(4,:) rate constant for the decay of organic P to dissolved P in reach at 20 deg C (1/day or 1/hour)*
- real \*8, dimension(:), allocatable [ammonian](#)  
*ammonia concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [orig\\_sedpstconc](#)
- real \*8, dimension(:,:), allocatable [wurch](#)  
*average daily water removal from the reach for the month (10<sup>4</sup> m<sup>3</sup>/day)*
- integer, dimension(:), allocatable [icanal](#)
- integer, dimension(:), allocatable [itb](#)
- real \*8, dimension(:), allocatable [ch\\_revap](#)  
*revap coeff: this variable controls the amount of water moving from bank storage to the root zone as a result of soil moisture depletion (none)*
- real \*8, dimension(:), allocatable [dep\\_chan](#)
- real \*8, dimension(:), allocatable [harg\\_petco](#)  
*coefficient related to radiation used in hargreaves eq (range: 0.0019 - 0.0032)*
- real \*8, dimension(:), allocatable [subfr\\_nowtr](#)
- real \*8, dimension(:), allocatable [cncoef\\_sub](#)  
*soil water depletion coefficient used in the new (modified curve number method) same as soil index coeff used in APEX range: 0.5 - 2.0*
- real \*8, dimension(:), allocatable [dr\\_sub](#)
- real \*8, dimension(:), allocatable [sub\\_fr](#)  
*fraction of total watershed area contained in subbasin (km2/km2)*

- real \*8, dimension(:), allocatable [sub\\_sw](#)  
*amount of water in soil profile on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable **wcklsp**
- real \*8, dimension(:), allocatable [sub\\_gwno3](#)  
*nitrate loading in groundwater from subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [sub\\_sumfc](#)  
*amount of water in soil at field capacity in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable **sub\_gwsolp**
- real \*8, dimension(:), allocatable [co2](#)  
*CO2 concentration (ppmv)*
- real \*8, dimension(:), allocatable [sub\\_km](#)  
*area of subbasin in square kilometers (km<sup>2</sup>)*
- real \*8, dimension(:), allocatable [sub\\_tc](#)  
*time of concentration for subbasin (hour)*
- real \*8, dimension(:), allocatable [sub\\_pet](#)  
*potential evapotranspiration for day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [welev](#)  
*elevation of weather station used to compile weather generator data (m)*
- real \*8, dimension(:), allocatable [sub\\_bd](#)  
*average bulk density in subbasin for top 10 mm of first soil layer (Mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable [sub\\_orgn](#)  
*amount of nitrogen stored in all organic pools in soil of subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [sub\\_orgp](#)  
*amount of phosphorus stored in all organic pools in soil of subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [sub\\_sedpa](#)  
*amount of active mineral P attached to sediment removed in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [sub\\_sedps](#)  
*amount of stable mineral P attached to sediment removed in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable **sub\_wtmp**
- real \*8, dimension(:), allocatable [daylmn](#)  
*shortest daylength occurring during the year (hour)*
- real \*8, dimension(:), allocatable [sub\\_minpa](#)  
*amount of phosphorus stored in active mineral pools sorbed to sediment (kg P/ha)*
- real \*8, dimension(:), allocatable [sub\\_minps](#)  
*amount of phosphorus stored in stable mineral pools sorbed to sediment (kg P/ha)*
- real \*8, dimension(:), allocatable [latcos](#)  
*cos(latitude) (none)*
- real \*8, dimension(:), allocatable [latsin](#)  
*sin(latitude) (none)*
- real \*8, dimension(:), allocatable [phutot](#)  
*total potential heat units for year (used when no crop is growing) (heat unit)*
- real \*8, dimension(:), allocatable [plaps](#)  
*precipitation lapse rate: precipitation change due to change in elevation (mm H2O/km)*
- real \*8, dimension(:), allocatable [tlaps](#)  
*temperature lapse rate: temperature change due to change in elevation (deg C/km)*
- real \*8, dimension(:), allocatable [tmp\\_an](#)  
*average annual air temperature (deg C)*
- real \*8, dimension(:), allocatable [sub\\_precip](#)  
*effective precipitation (amount of water reaching soil surface) for the day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [rammo\\_sub](#)  
*atmospheric deposition of ammonium values for entire watershed (mg/l)*

- real \*8, dimension(:), allocatable [rcn\\_sub](#)  
*atmospheric deposition of nitrate for entire watershed (mg/l)*
- real \*8, dimension(:), allocatable **pcpdays**
- real \*8, dimension(:), allocatable [sub\\_snom](#)  
*amount of snow melt in subbasin on day (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_qd](#)  
*surface runoff that reaches main channel during day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_sedy](#)  
*sediment yield for the day in subbasin (metric tons)*
- real \*8, dimension(:), allocatable [sub\\_tran](#)  
*transmission losses on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_no3](#)  
*NO3-N in surface runoff on day in subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [sub\\_latno3](#)  
*NO3-N in lateral flow on day in subbasin (kg N/ha)*
- real \*8, dimension(:,:), allocatable [sub\\_sftmp](#)  
*snowfall temperature for subbasin(:). Mean air temperature at which precip is equally likely to be rain as snow/freezing rain (range: -5.0/5.0) (deg C)*
- real \*8, dimension(:,:), allocatable [sub\\_smtmp](#)  
*snow melt base temperature for subbasin(:) mean air temperature at which snow melt will occur (range: -5.0/5.0) (deg C)*
- real \*8, dimension(:,:), allocatable [sub\\_timp](#)  
*snow pack temperature lag factor (0-1) (none)*  
*1 = no lag (snow pack temp=current day air temp) as the lag factor goes to zero, the snow pack's temperature will be less influenced by the current day's air temperature*
- real \*8, dimension(:), allocatable [sub\\_tileno3](#)  
*NO3 in tile flow on day in subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [sub\\_etday](#)  
*actual evapotranspiration on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_solp](#)  
*soluble P in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [sub\\_subp](#)  
*precipitation for day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_elev](#)  
*average elevation of HRU (m)*
- real \*8, dimension(:), allocatable [sub\\_surfq](#)  
*surface runoff generated on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_wyld](#)  
*water yield on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable **qird**
- real \*8, dimension(:), allocatable [sub\\_gwq](#)  
*groundwater flow on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_sep](#)  
*seepage from bottom of soil profile on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [sub\\_chl](#)  
*chlorophyll-a in water yield on day in subbasin (kg chl-a)*
- real \*8, dimension(:), allocatable [sub\\_cbod](#)  
*carbonaceous biological oxygen demand loading on day for subbasin (kg O2)*
- real \*8, dimension(:), allocatable [sub\\_dox](#)  
*dissolved oxygen loading on day for subbasin (kg O2)*
- real \*8, dimension(:), allocatable [sub\\_solpst](#)

- pesticide in solution in surface runoff on day in subbasin (mg pst)*

  - real \*8, dimension(:), allocatable [sub\\_yorgn](#)
- organic N loading in surface runoff on day in subbasin (kg P/ha)*

  - real \*8, dimension(:), allocatable [sub\\_yorgp](#)
- organic P loading in surface runoff on day in subbasin (kg P/ha)*

  - real \*8, dimension(:), allocatable [sub\\_sorpst](#)
- pesticide sorbed to sediment in surface runoff on day in subbasin (mg pst)*

  - real \*8, dimension(:), allocatable [sub\\_lat](#)
- latitude of HRU/subbasin (degrees)*

  - real \*8, dimension(:), allocatable [sub\\_bactlp](#)
- less persistent bacteria in surface runoff for day in subbasin (# cfu/m<sup>2</sup>)*

  - real \*8, dimension(:), allocatable [sub\\_bactp](#)
- persistent bacteria in surface runoff for day in subbasin (# cfu/m<sup>2</sup>)*

  - real \*8, dimension(:), allocatable [sub\\_latq](#)
  - real \*8, dimension(:), allocatable [sub\\_gwq\\_d](#)
  - real \*8, dimension(:), allocatable [sub\\_tileq](#)
  - real \*8, dimension(:), allocatable [sub\\_vaptile](#)
  - real \*8, dimension(:), allocatable [sub\\_dsan](#)
  - real \*8, dimension(:), allocatable [sub\\_dsil](#)
  - real \*8, dimension(:), allocatable [sub\\_dcla](#)
  - real \*8, dimension(:), allocatable [sub\\_dsag](#)
  - real \*8, dimension(:), allocatable [sub\\_dlag](#)
  - real \*8 [vap\\_tile](#)
  - real \*8, dimension(:, :), allocatable [sol\\_stpwt](#)
  - real \*8, dimension(:, :), allocatable [sub\\_hhwtmp](#)
- water temperature for the time step in subbasin (deg C)*

  - real \*8, dimension(:, :), allocatable [sub\\_hhqd](#)
  - real \*8, dimension(:, :), allocatable [huminc](#)
- monthly humidity adjustment. Daily values for relative humidity within the month are rasied or lowered by the specified amount (used in climate change studies) (none)*

  - real \*8, dimension(:, :), allocatable [radinc](#)
- monthly solar radiation adjustment. Daily radiation within the month is raised or lowered by the specified amount (used in climate change studies) (MJ/m<sup>2</sup>)*

  - real \*8, dimension(:, :), allocatable [rfinc](#)
- monthly rainfall adjustment. Daily rainfall within the month is adjusted to the specified percentage of the original value (used in climate change studies)(%)*

  - real \*8, dimension(:, :), allocatable [tmpinc](#)
- monthly temperature adjustment. Daily maximum and minimum temperatures within the month are raised or lowered by the specified amount (used in climate change studies) (deg C)*

  - real \*8, dimension(:, :), allocatable [ch\\_k](#)
- ch\_k(1,:) effective hydraulic conductivity of tributary channel alluvium (mm/hr)*  
*ch\_k(2,:) effective hydraulic conductivity of main channel alluvium (mm/hr)*

  - real \*8, dimension(:, :), allocatable [elevb](#)
- elevation at the center of the band in subbasin (m)*

  - real \*8, dimension(:, :), allocatable [elevb\\_fr](#)
- fraction of subbasin area within elevation band (the same fractions should be listed for all HRUs within the subbasin) (none)*

  - real \*8, dimension(:, :), allocatable [wndav](#)
- average wind speed for the month (m/s)*

  - real \*8, dimension(:, :), allocatable [ch\\_n](#)
- ch\_n(1,:) Manning's "n" value for the tributary channels (none)*  
*ch\_n(2,:) Manning's "n" value for the main channel (none)*

  - real \*8, dimension(:, :), allocatable [ch\\_s](#)

- ch\_s(1,:) average slope of tributary channels (m/m)*
- ch\_s(2,:) average slope of main channel (m/m)*
- real \*8, dimension(:,:), allocatable **ch\_w**
  - ch\_w(1,:) average width of tributary channels (m)*
  - ch\_w(2,:) average width of main channel (m)*
- real \*8, dimension(:,:), allocatable **dewpt**
  - average dew point temperature for the month (deg C)*
- real \*8, dimension(:,:), allocatable **amp\_r**
  - average fraction of total daily rainfall occurring in maximum half-hour period for month (none)*
- real \*8, dimension(:,:), allocatable **solarav**
  - average daily solar radiation for the month (MJ/m<sup>2</sup>/day)*
- real \*8, dimension(:,:), allocatable **tmpstdmx**
  - standard deviation for avg monthly maximum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **pcf**
  - normalization coefficient for precipitation generated from skewed distribution (none)*
- real \*8, dimension(:,:), allocatable **tmpmn**
  - avg monthly minimum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **tmpmx**
  - avg monthly maximum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **tmpstdmn**
  - standard deviation for avg monthly minimum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **otmpstdmn**
- real \*8, dimension(:,:), allocatable **otmpmn**
- real \*8, dimension(:,:), allocatable **otmpmx**
- real \*8, dimension(:,:), allocatable **otmpstdmx**
- real \*8, dimension(:,:), allocatable **ch\_erodmo**
- real \*8, dimension(:,:), allocatable **uh**
- real \*8, dimension(:,:), allocatable **hqdsave**
- real \*8, dimension(:,:), allocatable **hsdsave**
- real \*8, dimension(:,:), allocatable **pr\_w**
  - pr\_w(1,:) probability of wet day after dry day in month (none)*
  - pr\_w(2,:) probability of wet day after wet day in month (none)*
  - pr\_w(3,:) proportion of wet days in the month (none)*
- real \*8, dimension(:,:), allocatable **pcp\_stat**
- real \*8, dimension(:,:), allocatable **opr\_w**
- real \*8, dimension(:,:), allocatable **opcp\_stat**
- integer, dimension(:), allocatable **ireg**
  - precipitation category (none):*
  - 1 precipitation <= 508 mm/yr*
  - 2 precipitation > 508 and <= 1016 mm/yr*
  - 3 precipitation > 1016 mm/yr*
- integer, dimension(:), allocatable **hrutot**
  - number of HRUs in subbasin (none)*
- integer, dimension(:), allocatable **hru1**
- integer, dimension(:), allocatable **ihgage**
  - HRU relative humidity data code (gage # for relative humidity data used in as HRU) (none)*
- integer, dimension(:), allocatable **isgage**
  - HRU solar radiation data code (record # for solar radiation used in HRU) (none)*
- integer, dimension(:), allocatable **iwgage**
  - HRU wind speed gage data code (gage # for wind speed data used in HRU) (none)*
- integer, dimension(:), allocatable **subgis**
  - GIS code printed to output files (output.sub, .rch) (none)*
- integer, dimension(:), allocatable **irgage**



- subbasin rain gage data code (gage # for rainfall data used in HRU) (none)*
- integer, dimension(:), allocatable [itgage](#)
  - subbasin temp gage data code (gage # for temperature data used in HRU) (none)*
- integer, dimension(:), allocatable [irelh](#)
  - (none) irelh = 0 (dewpoint)*
  - irelh = 1 (relative humidity)*
  - note: inputs > 1.0 (dewpoint)*
  - inputs < 1.0 (relative hum)*
- integer, dimension(:), allocatable [fcst\\_reg](#)
- real \*8, dimension(:,:), allocatable [sol\\_aorgn](#)
  - amount of nitrogen stored in the active organic (humic) nitrogen pool in soil layer (kg N/ha)*
- real \*8, dimension(:,:), allocatable [sol\\_fon](#)
  - amount of nitrogen stored in the fresh organic (residue) pool in soil layer (kg N/ha)*
- real \*8, dimension(:,:), allocatable [sol\\_tmp](#)
  - average temperature of soil layer on previous day or*
  - daily average temperature of soil layer (deg C)*
- real \*8, dimension(:,:), allocatable [sol\\_awc](#)
  - available water capacity of soil layer (mm H2O/mm soil)*
- real \*8, dimension(:,:), allocatable [volcr](#)
  - crack volume for soil layer (mm)*
- real \*8, dimension(:,:), allocatable [sol\\_prk](#)
  - percolation storage from soil layer on current day (mm H2O)*
- real \*8, dimension(:,:), allocatable [pperco\\_sub](#)
  - subbasin phosphorus percolation coefficient. Ratio of soluble phosphorus in surface to soluble phosphorus in percolate*
- real \*8, dimension(:,:), allocatable [sol\\_stap](#)
  - amount of phosphorus in the soil layer stored in the stable mineral phosphorus pool (kg P/ha)*
- real \*8, dimension(:,:), allocatable [conv\\_wt](#)
  - factor which converts kg/kg soil to kg/ha (none)*
- real \*8, dimension(:,:), allocatable [sol\\_actp](#)
  - amount of phosphorus stored in the active mineral phosphorus pool (kg P/ha)*
- real \*8, dimension(:,:), allocatable [sol\\_solp](#)
  - soluble P concentration in top soil layer (mg P/kg soil) or*
  - amount of inorganic phosphorus stored in solution in soil layer. NOTE UNIT CHANGE! (kg P/ha)*
- real \*8, dimension(:,:), allocatable [crdep](#)
  - maximum or potential crack volume (mm)*
- real \*8, dimension(:,:), allocatable [sol\\_fc](#)
  - amount of water available to plants in soil layer at field capacity (fc - wp water) (mm H2O)*
- real \*8, dimension(:,:), allocatable [sol\\_ul](#)
  - amount of water held in the soil layer at saturation (sat - wp water) (mm H2O)*
- real \*8, dimension(:,:), allocatable [sol\\_bd](#)
  - bulk density of the soil layer in HRU (Mg/m<sup>3</sup>)*
- real \*8, dimension(:,:), allocatable [sol\\_z](#)
  - depth to bottom of each soil profile layer in a given HRU (mm)*
- real \*8, dimension(:,:), allocatable [sol\\_st](#)
  - amount of water stored in the soil layer on any given day (less wilting point water) (mm H2O)*
- real \*8, dimension(:,:), allocatable [sol\\_up](#)
  - water content of soil at -0.033 MPa (field capacity) (mm H2O/mm soil)*
- real \*8, dimension(:,:), allocatable [sol\\_clay](#)
  - percent clay content in soil layer in HRU (UNIT CHANGE!) (% or none)*
- real \*8, dimension(:,:), allocatable [sol\\_hk](#)
  - beta coefficient to calculate hydraulic conductivity (none)*



- real \*8, dimension(:,:), allocatable **flat**  
*lateral flow storage in soil layer on current day (mm H2O)*
- real \*8, dimension(:,:), allocatable **sol\_nh3**  
*amount of nitrogen stored in the ammonium pool in soil layer (kg N/ha)*
- real \*8, dimension(:,:), allocatable **sol\_ec**  
*electrical conductivity of soil layer (dS/m)*
- real \*8, dimension(:,:), allocatable **sol\_orgn**  
*amount of nitrogen stored in the stable organic N pool. NOTE UNIT CHANGE! (mg N/kg soil or kg N/ha)*
- real \*8, dimension(:,:), allocatable **sol\_por**  
*total porosity of soil layer expressed as a fraction of the total volume (none)*
- real \*8, dimension(:,:), allocatable **sol\_wp**  
*water content of soil at -1.5 MPa (wilting point) (mm H2O/mm soil)*
- real \*8, dimension(:,:), allocatable **sol\_orgp**  
*amount of phosphorus stored in the organic P pool in soil layer. NOTE UNIT CHANGE! (mg P/kg soil or kg P/ha)*
- real \*8, dimension(:,:), allocatable **sol\_wpmm**  
*water content of soil at -1.5 MPa (wilting point) (mm H2O)*
- real \*8, dimension(:,:), allocatable **sol\_no3**  
*amount of nitrogen stored in the nitrate pool in the soil layer. This variable is read in as a concentration and converted to kg/ha (this value is read from the .sol file in units of mg/kg) (kg N/ha)*
- real \*8, dimension(:,:), allocatable **sol\_cbn**  
*percent organic carbon in soil layer (%)*
- real \*8, dimension(:,:), allocatable **sol\_k**  
*saturated hydraulic conductivity of soil layer (mm/hour)*
- real \*8, dimension(:,:), allocatable **sol\_rsd**  
*amount of organic matter in the soil layer classified as residue (kg/ha)*
- real \*8, dimension(:,:), allocatable **sol\_fop**  
*amount of phosphorus stored in the fresh organic (residue) pool in soil layer (kg P/ha)*
- real \*8, dimension(:,:), allocatable **sol\_rock**  
*percent of rock fragments in soil layer (%)*
- real \*8, dimension(:,:), allocatable **sol\_silt**  
*percent silt content in soil material (UNIT CHANGE!) (%) or none)*
- real \*8, dimension(:,:), allocatable **sol\_sand**  
*percent sand content of soil material (%)*
- real \*8, dimension(:,:), allocatable **orig\_solno3**
- real \*8, dimension(:,:), allocatable **orig\_solorgn**
- real \*8, dimension(:,:), allocatable **orig\_solsolp**
- real \*8, dimension(:,:), allocatable **orig\_solorgp**
- real \*8, dimension(:,:), allocatable **orig\_soltmp**
- real \*8, dimension(:,:), allocatable **orig\_solrsd**
- real \*8, dimension(:,:), allocatable **orig\_solfop**
- real \*8, dimension(:,:), allocatable **orig\_solfon**
- real \*8, dimension(:,:), allocatable **orig\_solaorgn**
- real \*8, dimension(:,:), allocatable **orig\_solst**
- real \*8, dimension(:,:), allocatable **orig\_solactp**
- real \*8, dimension(:,:), allocatable **orig\_solstap**
- real \*8, dimension(:,:), allocatable **orig\_volcr**
- real \*8, dimension(:,:), allocatable **sol\_pst**  
*sol\_pst(:,1) initial amount of pesticide in first layer read in from .chm file (mg/kg)*  
*sol\_pst(:,,:) amount of pesticide in soil layer. NOTE UNIT CHANGE! (kg/ha)*
- real \*8, dimension(:,:), allocatable **sol\_kp**  
*pesticide sorption coefficient, Kp; the ratio of the concentration in the solid phase to the concentration in solution ((mg/kg)/(mg/L) or m<sup>3</sup>/ton)*

- real \*8, dimension(:, :, :), allocatable **orig\_solpst**
- real \*8, dimension(:), allocatable **velsetlr**
- real \*8, dimension(:), allocatable **velsetlp**
- real \*8, dimension(:, :), allocatable **br**
  - br(1,:) 1st shape parameter for reservoir surface area equation (none)*
  - br(2,:) 2nd shape parameter for reservoir surface area equation (none)*
- real \*8, dimension(:), allocatable **evrsv**
  - lake evaporation coefficient (none)*
- real \*8, dimension(:), allocatable **res\_k**
  - hydraulic conductivity of the reservoir bottom (mm/hr)*
- real \*8, dimension(:), allocatable **lkpst\_conc**
  - pesticide concentration in lake water (mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **res\_evol**
  - volume of water needed to fill the reservoir to the emergency spillway (read in as 10<sup>4</sup> m<sup>3</sup> and converted to m<sup>3</sup>) (m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **res\_pvol**
  - volume of water needed to fill the reservoir to the principal spillway (read in as 10<sup>4</sup> m<sup>3</sup> and converted to m<sup>3</sup>) (m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **res\_vol**
  - reservoir volume (read in as 10<sup>4</sup> m<sup>3</sup> and converted to m<sup>3</sup>) (m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **res\_psa**
  - reservoir surface area when reservoir is filled to principal spillway (ha)*
- real \*8, dimension(:), allocatable **lkpst\_rea**
  - pesticide reaction coefficient in lake water (1/day)*
- real \*8, dimension(:), allocatable **lkpst\_vol**
  - pesticide volatilization coefficient in lake water (m/day)*
- real \*8, dimension(:), allocatable **res\_rr**
  - average daily principal spillway release volume (read in as a release rate in m<sup>3</sup>/s and converted to m<sup>3</sup>/day) (m<sup>3</sup>/day)*
- real \*8, dimension(:), allocatable **res\_sed**
  - amount of sediment in reservoir (read in as mg/L and converted to kg/L) (kg/L)*
- real \*8, dimension(:), allocatable **lkpst\_koc**
  - pesticide partition coefficient between water and sediment in lake water (m<sup>3</sup>/g)*
- real \*8, dimension(:), allocatable **lkpst\_mix**
  - mixing velocity (diffusion/dispersion) in lake water for pesticide (m/day)*
- real \*8, dimension(:), allocatable **lkpst\_rsp**
  - resuspension velocity in lake water for pesticide sorbed to sediment (m/day)*
- real \*8, dimension(:), allocatable **lkpst\_stl**
  - settling velocity in lake water for pesticide sorbed to sediment (m/day)*
- real \*8, dimension(:), allocatable **lkspst\_conc**
  - pesticide concentration in lake bed sediment (mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **lkspst\_rea**
  - pesticide reaction coefficient in lake bed sediment (1/day)*
- real \*8, dimension(:), allocatable **theta\_n**
- real \*8, dimension(:), allocatable **theta\_p**
- real \*8, dimension(:), allocatable **con\_nirr**
- real \*8, dimension(:), allocatable **con\_pirr**
- real \*8, dimension(:), allocatable **lkspst\_act**
  - depth of active sediment layer in lake for for pesticide (m)*
- real \*8, dimension(:), allocatable **lkspst\_bry**
  - pesticide burial velocity in lake bed sediment (m/day)*
- real \*8, dimension(:), allocatable **sed\_stlr**

- real \*8, dimension(7) **resdata**  
*resdata(1) average annual evaporation from reservoirs in watershed (mm H2O)*  
*resdata(2) average annual seepage from reservoirs in watershed (mm H2O)*  
*resdata(3) average annual precipitation on reservoirs in watershed (mm H2O)*  
*resdata(4) average annual amount of water transported into reservoirs in watershed (mm H2O)*  
*resdata(5) average annual amount of sediment transported into reservoirs in watershed (metric tons/ha)*  
*resdata(6) average annual amount of water transported out of reservoirs in watershed (mm H2O)*  
*resdata(7) average annual amount of sediment transported out of reservoirs in watershed (metric tons/ha)*
- real \*8, dimension(:), allocatable **res\_nsed**  
*normal amount of sediment in reservoir (read in as mg/L and convert to kg/L) (kg/L)*
- real \*8, dimension(:), allocatable **wurtmf**  
*fraction of water removed from the reservoir via WURESN which is returned and becomes flow from the reservoir outlet (none)*
- real \*8, dimension(:), allocatable **chlar**  
*chlorophyll-a production coefficient for reservoir (none)*
- real \*8, dimension(:), allocatable **res\_no3**  
*amount of nitrate in reservoir (kg N)*
- real \*8, dimension(:), allocatable **res\_orgn**  
*amount of organic N in reservoir (kg N)*
- real \*8, dimension(:), allocatable **res\_orgp**  
*amount of organic P in reservoir (kg P)*
- real \*8, dimension(:), allocatable **res\_solp**  
*amount of soluble P in reservoir (kg P)*
- real \*8, dimension(:), allocatable **res\_seci**  
*secchi-disk depth (m)*
- real \*8, dimension(:), allocatable **res\_nh3**  
*amount of ammonia in reservoir (kg N)*
- real \*8, dimension(:), allocatable **res\_no2**  
*amount of nitrite in reservoir (kg N)*
- real \*8, dimension(:), allocatable **seccir**  
*water clarity coefficient for reservoir (none)*
- real \*8, dimension(:), allocatable **oflowmn\_fps**  
*minimum reservoir outflow as a fraction of the principal spillway volume (0-1) (fraction)*
- real \*8, dimension(:), allocatable **starg\_fps**  
*target volume as a fraction of the principal spillway volume (.1-5) (fraction)*
- real \*8, dimension(:), allocatable **weirc**
- real \*8, dimension(:), allocatable **weirk**
- real \*8, dimension(:), allocatable **weirw**
- real \*8, dimension(:), allocatable **acoef**
- real \*8, dimension(:), allocatable **bcoef**
- real \*8, dimension(:), allocatable **ccoef**
- real \*8, dimension(:), allocatable **orig\_resvol**
- real \*8, dimension(:), allocatable **orig\_ressed**
- real \*8, dimension(:), allocatable **orig\_lkpstconc**
- real \*8, dimension(:), allocatable **orig\_lkspstconc**
- real \*8, dimension(:), allocatable **orig\_ressolp**
- real \*8, dimension(:), allocatable **orig\_resorgp**
- real \*8, dimension(:), allocatable **orig\_resno3**
- real \*8, dimension(:), allocatable **orig\_resno2**
- real \*8, dimension(:), allocatable **orig\_resnh3**
- real \*8, dimension(:), allocatable **orig\_resorgn**
- real \*8, dimension(:,:), allocatable **oflowmn**  
*minimum daily outflow for the month (read in as m<sup>3</sup>/s and converted to m<sup>3</sup>/day) (m<sup>3</sup>/day)*

- real \*8, dimension(:,:), allocatable **oflowmx**  
*maximum daily outflow for the month (read in as  $m^3/s$  and converted to  $m^3/day$ ) ( $m^3/day$ )*
- real \*8, dimension(:,:), allocatable **starg**  
*monthly target reservoir storage (needed if IRESKO=2) (read in as  $10^4 m^3$  and converted to  $m^3$ ) ( $m^3$ )*
- real \*8, dimension(:,:), allocatable **psetlr**  
*psetlr(1,:) phosphorus settling rate for mid-year period (read in as  $m/year$  and converted to  $m/day$ ) ( $m/day$ )*  
*psetlr(2,:) phosphorus settling rate for remainder of year (read in as  $m/year$  and converted to  $m/day$ ) ( $m/day$ )*
- real \*8, dimension(:,:), allocatable **nsetlr**  
*nsetlr(1,:) nitrogen settling rate for mid-year period (read in as  $m/year$  and converted to  $m/day$ ) ( $m/day$ )*  
*nsetlr(2,:) nitrogen settling rate for remainder of year (read in as  $m/year$  and converted to  $m/day$ ) ( $m/day$ )*
- real \*8, dimension(:,:), allocatable **wuresn**  
*average amount of water withdrawn from reservoir each month for consumptive water use (read in as  $10^4 m^3$  and converted to  $m^3$ ) ( $m^3$ )*
- real \*8, dimension(:,:), allocatable **res\_out**  
*measured average daily outflow from the reservoir for the month (needed if IRESKO=1) (read in as  $m^3/s$  and converted to  $m^3/day$ ) ( $m^3/day$ )*
- integer, dimension(:), allocatable **res\_sub**  
*number of subbasin reservoir is in (weather for the subbasin is used for the reservoir) (none)*
- integer, dimension(:,:), allocatable **ires**  
*ires(1,:) beginning of mid-year nutrient settling "season" (none)*  
*ires(2,:) end of mid-year nutrient settling "season" (none)*
- integer, dimension(:), allocatable **iresco**  
*outflow simulation code (none):*  
*0 compute outflow for uncontrolled reservoir with average annual release rate*  
*1 measured monthly outflow*  
*2 simulated controlled outflow-target release*  
*3 measured daily outflow*  
*4 stage/volume/outflow relationship*
- integer, dimension(:), allocatable **iyres**  
*year of the simulation that the reservoir becomes operational (none)*
- integer, dimension(:), allocatable **mores**  
*month the reservoir becomes operational (none)*
- integer, dimension(:,:), allocatable **iflodr**  
*iflodr(1,:) beginning month of non-flood season (needed if IRESKO=2) (none)*  
*iflodr(2,:) ending month of non-flood season (needed if IRESKO=2) (none)*
- integer, dimension(:), allocatable **ndtargr**  
*number of days to reach target storage from current reservoir storage (needed if IRESKO=2) (days)*
- real \*8, dimension(:), allocatable **ap\_ef**  
*application efficiency (0-1) (none)*
- real \*8, dimension(:), allocatable **decay\_f**  
*exponential of the rate constant for degradation of the pesticide on foliage (none)*
- real \*8, dimension(:), allocatable **skoc**  
*soil adsorption coefficient normalized for soil organic carbon content  $((mg/kg)/(mg/L))$*
- real \*8, dimension(:), allocatable **decay\_s**  
*exponential of the rate constant for degradation of the pesticide in soil (none)*
- real \*8, dimension(:), allocatable **pst\_wof**  
*fraction of pesticide on foliage which is washed-off by a rainfall event (none)*
- real \*8, dimension(:), allocatable **pst\_wsol**  
*solubility of chemical in water ( $mg/L$  ( $ppm$ ))*
- real \*8, dimension(:), allocatable **irramt**  
*depth of irrigation water applied to HRU ( $mm H_2O$ )*
- real \*8, dimension(:), allocatable **phusw**
- integer, dimension(:), allocatable **pstflg**

- flag for types of pesticide used in watershed. Array location is pesticide ID number*  
*0: pesticide not used*  
*1: pesticide used*
- integer, dimension(:), allocatable **nope**  
*sequence number of pesticide in NPNO(:) (none)*
  - integer, dimension(:), allocatable **nop**
  - integer, dimension(:), allocatable **yr\_skip**
  - integer, dimension(:), allocatable **icrmx**
  - integer, dimension(:), allocatable **nopmx**
  - integer, dimension(:,:), allocatable **mgtop**
  - integer, dimension(:,:), allocatable **idop**
  - integer, dimension(:,:), allocatable **mgt1iop**
  - integer, dimension(:,:), allocatable **mgt2iop**
  - integer, dimension(:,:), allocatable **mgt3iop**
  - integer, dimension(:,:), allocatable **mgt10iop**
  - real \*8, dimension(:,:), allocatable **mgt4op**
  - real \*8, dimension(:,:), allocatable **mgt5op**
  - real \*8, dimension(:,:), allocatable **mgt6op**
  - real \*8, dimension(:,:), allocatable **mgt7op**
  - real \*8, dimension(:,:), allocatable **mgt8op**
  - real \*8, dimension(:,:), allocatable **mgt9op**
  - real \*8, dimension(:,:), allocatable **phu\_op**
  - real \*8, dimension(:), allocatable **cnyld**  
*fraction of nitrogen in yield (kg N/kg yield)*
  - real \*8, dimension(:), allocatable **rsdco\_pl**  
*plant residue decomposition coefficient. The fraction of residue which will decompose in a day assuming optimal moisture, temperature, C:N ratio, and C:P ratio (none)*
  - real \*8, dimension(:,:), allocatable **wac2**  
*wac2(1,:) 1st shape parameter for radiation use efficiency equation (none)*  
*wac2(2,:) 2nd shape parameter for radiation use efficiency equation (none)*
  - real \*8, dimension(:), allocatable **alai\_min**  
*minimum LAI during winter dormant period ( $m^2/m^2$ )*
  - real \*8, dimension(:,:), allocatable **leaf**  
*leaf(1,:) 1st shape parameter for leaf area development equation (none)*  
*leaf(2,:) 2nd shape parameter for leaf area development equation (none)*
  - real \*8, dimension(:), allocatable **wsyf**  
*Value of harvest index between 0 and HVSTI which represents the lowest value expected due to water stress ((kg/ha)/(kg/ha))*
  - real \*8, dimension(:), allocatable **bio\_e**  
*biomass-energy ratio. The potential (unstressed) growth rate per unit of intercepted photosynthetically active radiation ((kg/ha)/(MJ/m\*2))*
  - real \*8, dimension(:), allocatable **hvsti**  
*harvest index: crop yield/aboveground biomass ((kg/ha)/(kg/ha))*
  - real \*8, dimension(:), allocatable **t\_base**  
*minimum temperature for plant growth (deg C)*
  - real \*8, dimension(:), allocatable **t\_opt**  
*optimal temperature for plant growth (deg C)*
  - real \*8, dimension(:), allocatable **chtmx**  
*maximum canopy height (m)*
  - real \*8, dimension(:), allocatable **cvm**  
*natural log of USLE\_C (the minimum value of the USLE C factor for the land cover) (none)*
  - real \*8, dimension(:), allocatable **gsi**  
*maximum stomatal conductance (m/s)*

- real \*8, dimension(:), allocatable **vpd2**  
*rate of decline in stomatal conductance per unit increase in vapor pressure deficit ((m/s)\*(1/kPa))*
- real \*8, dimension(:), allocatable **wavp**  
*rate of decline in radiation use efficiency as a function of vapor pressure deficit (none)*
- real \*8, dimension(:), allocatable **bio\_leaf**  
*fraction of leaf/needle biomass that drops during dormancy (for trees only) (none)*
- real \*8, dimension(:), allocatable **blai**  
*maximum (potential) leaf area index (none)*
- real \*8, dimension(:), allocatable **cpyld**  
*fraction of phosphorus in yield (kg P/kg yield)*
- real \*8, dimension(:), allocatable **dlai**  
*fraction of growing season when leaf area declines (none)*
- real \*8, dimension(:), allocatable **rdmx**  
*maximum root depth of plant (m)*
- real \*8, dimension(:, :), allocatable **bio\_n**  
*bio\_n(1,:) 1st shape parameter for plant N uptake equation (none)*  
*bio\_n(2,:) 2nd shape parameter for plant N uptake equation (none)*
- real \*8, dimension(:, :), allocatable **bio\_p**  
*bio\_p(1,:) 1st shape parameter for plant P uptake equation (none)*  
*bio\_p(2,:) 2nd shape parameter for plant P uptake equation (none)*
- real \*8, dimension(:), allocatable **bm\_dieoff**  
*fraction above ground biomass that dies off at dormancy (fraction)*
- real \*8, dimension(:), allocatable **bm\_x\_trees**
- real \*8, dimension(:), allocatable **ext\_coef**
- real \*8, dimension(:, :), allocatable **rsr**  
*rsr(1,:) initial root to shoot ratio at the beg of growing season*  
*rsr(2,:) root to shoot ratio at the end of the growing season*
- real \*8, dimension(:), allocatable **pltnfr1**  
*nitrogen uptake parameter #1: normal fraction of N in crop biomass at emergence (kg N/kg biomass)*
- real \*8, dimension(:), allocatable **pltnfr3**  
*nitrogen uptake parameter #3: normal fraction of N in crop biomass at maturity (kg N/kg biomass)*
- real \*8, dimension(:), allocatable **pltpfr1**  
*phosphorus uptake parameter #1: normal fraction of P in crop biomass at emergence (kg P/kg biomass)*
- real \*8, dimension(:), allocatable **pltpfr3**  
*phosphorus uptake parameter #3: normal fraction of P in crop biomass at maturity (kg P/kg biomass)*
- integer, dimension(:), allocatable **idc**  
*crop/landcover category (none):*  
*1 warm season annual legume*  
*2 cold season annual legume*  
*3 perennial legume*  
*4 warm season annual*  
*5 cold season annual*  
*6 perennial*  
*7 trees*
- integer, dimension(:), allocatable **mat\_yrs**
- real \*8, dimension(:), allocatable **bactpdb**  
*concentration of persistent bacteria in manure (fertilizer) (cfu/g manure)*
- real \*8, dimension(:), allocatable **fminn**  
*fraction of fertilize/manure that is mineral nitrogen (NO<sub>3</sub> + NH<sub>3</sub>) (kg minN/kg fert)*
- real \*8, dimension(:), allocatable **forgn**  
*fraction of organic nitrogen in fertilizer/manure (kg orgN/kg fert)*
- real \*8, dimension(:), allocatable **forgp**

- fraction of fertilizer/manure that is organic phosphorus (kg orgP/kg fert)*

  - real \*8, dimension(:), allocatable [bactkddb](#)
- fraction of bacteria in solution (the remaining fraction is sorbed to soil particles) (none):*

*1: all bacteria in solution*

*0: all bacteria sorbed to soil particles*

  - real \*8, dimension(:), allocatable [bactlpdb](#)
- concentration of less persistent bacteria in manure (fertilizer) (cfu/g manure)*

  - real \*8, dimension(:), allocatable [fminp](#)
- fraction of fertilizer that is mineral phosphorus in fertilizer/manure (kg minP/kg fert)*

  - real \*8, dimension(:), allocatable [fnh3n](#)
- fraction of mineral N content that is NH3-N in fertilizer/manure (kg NH3-N/kg minN)*

  - character(len=8), dimension(200) [fertnm](#)
- name of fertilizer*

  - real \*8, dimension(:), allocatable [curbden](#)
- curb length density in HRU (km/ha)*

  - real \*8, dimension(:), allocatable [dirtmx](#)
- maximum amount of solids allowed to build up on impervious surfaces (kg/curb km)*

  - real \*8, dimension(:), allocatable [fimp](#)
- fraction of HRU area that is impervious (both directly and indirectly connected) (fraction)*

  - real \*8, dimension(:), allocatable [urbcoef](#)
- wash-off coefficient for removal of constituents from an impervious surface (1/mm)*

  - real \*8, dimension(:), allocatable [thalf](#)
- time for the amount of solids on impervious areas to build up to 1/2 the maximum level (days)*

  - real \*8, dimension(:), allocatable [tncon](#)
- concentration of total nitrogen in suspended solid load from impervious areas (mg N/kg sed)*

  - real \*8, dimension(:), allocatable [tno3conc](#)
- concentration of NO3-N in suspended solid load from impervious areas (mg NO3-N/kg sed)*

  - real \*8, dimension(:), allocatable [tpconc](#)
- concentration of total phosphorus in suspended solid load from impervious areas (mg P/kg sed)*

  - real \*8, dimension(:), allocatable [fcimp](#)
- fraction of HRU area that is classified as directly connected impervious (fraction)*

  - real \*8, dimension(:), allocatable [urbcn2](#)
- SCS curve number for moisture condition II in impervious areas (none)*

  - real \*8 [fr\\_curb](#)
- availability factor, the fraction of the curb length that is sweepable (none)*

  - real \*8 [frt\\_kg](#)
- amount of fertilizer applied to HRU (kg/ha)*

  - real \*8 [pst\\_dep](#)
- depth of pesticide in the soil (mm)*

  - real \*8 [swepeff](#)
- removal efficiency of sweeping operation (none)*

  - real \*8, dimension(:), allocatable [ranrns\\_hru](#)
- random roughness for a given HRU (mm)*

  - integer, dimension(:), allocatable [itill](#)
- depth of mixing caused by tillage operation (mm)*

  - real \*8, dimension(:), allocatable [deptil](#)
- mixing efficiency of tillage operation (none)*

  - real \*8, dimension(:), allocatable [effmix](#)
- random roughness of a given tillage operation (mm)*

  - real \*8, dimension(:), allocatable [ranrns](#)
- random roughness of a given tillage operation (mm)*

  - character(len=8), dimension(550) [tillnm](#)



- 8-character name for the tillage operation

  - real \*8, dimension(:), allocatable **num1s**
    - For ICODES equal to (none)*
    - 0,1,3,5,9: not used*
    - 2: fraction of overland flow in channel*
    - 4: amount of water transferred (as defined by INUM4S)*
    - 7,8,10,11: drainage area in square kilometers associated with the record file*
    - 12: reparation coefficient.*
  - real \*8, dimension(:), allocatable **hyd\_dakm**
    - total drainage area of hydrograph in square kilometers (km<sup>2</sup>)*
  - real \*8, dimension(:, :), allocatable **shyd**
    - shyd(1,:) water (m<sup>3</sup> H<sub>2</sub>O)*
    - shyd(2,:) sediment or suspended solid load (metric tons)*
    - shyd(3,:) organic nitrogen (kg N)*
    - shyd(4,:) organic phosphorus (kg P)*
    - shyd(5,:) nitrate (kg N)*
    - shyd(6,:) soluble phosphorus (kg P)*
    - shyd(7,:) soluble pesticides (kg P)*
    - shyd(8,:) sorbed pesticides (kg P)*
  - real \*8, dimension(:, :), allocatable **varoute**
    - varoute(:, :) daily routing storage array (varies):*
    - varoute(1,:) temperature (deg C)*
    - varoute(2,:) water (m<sup>3</sup> H<sub>2</sub>O)*
    - varoute(3,:) sediment or suspended solid load (metric tons)*
    - varoute(4,:) organic nitrogen (kg N)*
    - varoute(5,:) organic phosphorus (kg P)*
    - varoute(6,:) nitrate (kg N)*
    - varoute(7,:) soluble mineral phosphorus (kg P)*
    - varoute(11,:) pesticide in solution (mg pst)*
    - varoute(12,:) pesticide sorbed to sediment (mg pst)*
    - varoute(13,:) chlorophyll-a (kg)*
    - varoute(14,:) ammonium (kg N)*
    - varoute(15,:) nitrite (kg N)*
    - varoute(16,:) carbonaceous biological oxygen demand (kg)*
    - varoute(17,:) dissolved oxygen (kg)*
    - varoute(18,:) persistent bacteria (# cfu/100ml)*
    - varoute(19,:) less persistent bacteria (# cfu/100ml)*
    - varoute(20,:) conservative metal #1 (kg)*
    - varoute(21,:) conservative metal #2 (kg)*
    - varoute(22,:) conservative metal #3 (kg)*
  - real \*8, dimension(:, :), allocatable **vartran**
  - real \*8, dimension(:, :), allocatable **hhvaroute**
    - routing storage array for hourly time step (varies)*
    - hhvaroute(1,:) temperature (deg C)*
    - hhvaroute(2,:) water (m<sup>3</sup> H<sub>2</sub>O)*
    - hhvaroute(3,:) sediment or suspended solid load (metric tons)*
    - hhvaroute(4,:) organic nitrogen (kg N)*
    - hhvaroute(5,:) organic phosphorus (kg P)*
    - hhvaroute(6,:) nitrate (kg N)*
    - hhvaroute(7,:) soluble mineral phosphorus (kg P)*
    - hhvaroute(11,:) pesticide in solution (mg pst)*
    - hhvaroute(12,:) pesticide sorbed to sediment (mg pst)*
    - hhvaroute(13,:) chlorophyll-a (kg)*
    - hhvaroute(14,:) ammonium (kg N)*
    - hhvaroute(15,:) nitrite (kg N)*
    - hhvaroute(16,:) carbonaceous biological oxygen demand (kg)*
    - hhvaroute(17,:) dissolved oxygen (kg O<sub>2</sub>)*
    - hhvaroute(18,:) persistent bacteria (# cfu/100ml)*
    - hhvaroute(19,:) less persistent bacteria (# cfu/100ml)*
    - hhvaroute(20,:) conservative metal #1 (kg)*
    - hhvaroute(21,:) conservative metal #2 (kg)*
    - hhvaroute(22,:) conservative metal #3 (kg)*
  - integer, dimension(:), allocatable **icodes**



*routing command code (none):*

0 = finish  
 1 = subbasin  
 2 = route  
 3 = routes  
 4 = transfer  
 5 = add  
 6 = rehour  
 7 = recmon  
 8 = recyear  
 9 = save  
 10 = recday  
 11 = recnst  
 12 = structure  
 13 = apex  
 14 = saveconc  
 15 =  
 16 = autocal  
 17 = routing unit

- integer, dimension(:), allocatable **ihouts**

*For ICODES equal to (none)*

0: not used  
 1,2,3,5,6,7,8,10,11: hydrograph storage location number  
 4: departure type (1=reach, 2=reservoir)  
 9: hydrograph storage location of data to be printed to event file  
 14: hydrograph storage location of data to be printed to saveconc file.

- integer, dimension(:), allocatable **inum1s**

*For ICODES equal to (none)*

0: not used  
 1: subbasin number  
 2: reach number  
 3: reservoir number  
 4: reach or res # flow is diverted from  
 5: hydrograph storage location of 1st dataset to be added  
 6,7,8,9,10,11,14: file number.

- integer, dimension(:), allocatable **inum2s**

*For ICODES equal to (none)*

0,1,7,8,10,11: not used  
 2,3: inflow hydrograph storage location  
 4: destination type (1=reach, 2=reservoir)  
 5: hydrograph storage location of 2nd dataset to be added  
 9,14: print frequency (0=daily, 1=hourly)

- integer, dimension(:), allocatable **inum3s**

*For ICODES equal to (none)*

0,1,5,7,8,10,11: not used  
 2,3: subbasin number 4: destination number. Reach or reservoir receiving water  
 9: print format (0=normal, fixed format; 1=txt format for AV interface, recday)

- integer, dimension(:), allocatable **inum4s**

*For ICODES equal to (none)*

0,2,3,5,7,8,9,10,11: not used  
 1: GIS code printed to output file (optional)  
 4: rule code governing transfer of water (1=fraction transferred out, 2=min volume or flow left, 3=exact amount transferred)

- integer, dimension(:), allocatable **inum5s**
- integer, dimension(:), allocatable **inum6s**
- integer, dimension(:), allocatable **inum7s**
- integer, dimension(:), allocatable **inum8s**
- integer, dimension(:), allocatable **subed**
- character(len=10), dimension(:), allocatable **recmonps**
- character(len=10), dimension(:), allocatable **reccnstps**

- character(len=5), dimension(:), allocatable **subnum**
- character(len=4), dimension(:), allocatable **hruno**
- real \*8, dimension(:), allocatable **grwat\_n**  
*Mannings's n for grassed waterway (none)*
- integer, dimension(:), allocatable **grwat\_i**  
*flag for the simulation of grass waterways (none)*  
*= 0 inactive*  
*= 1 active*
- real \*8, dimension(:), allocatable **grwat\_l**  
*length of grass waterway (km)*
- real \*8, dimension(:), allocatable **grwat\_w**  
*average width of grassed waterway (m)*
- real \*8, dimension(:), allocatable **grwat\_d**  
*depth of grassed waterway from top of bank to bottom (m)*
- real \*8, dimension(:), allocatable **grwat\_s**  
*average slope of grassed waterway channel (m)*
- real \*8, dimension(:), allocatable **grwat\_spcon**  
*linear parameter defined by user for calculating sediment transport in grassed waterways (none)*
- real \*8, dimension(:), allocatable **tc\_gwat**  
*time of concentration for grassed waterway and its drainage area (none)*
- real \*8, dimension(:), allocatable **pot\_tilemm**
- real \*8, dimension(:), allocatable **pot\_volxmm**
- real \*8, dimension(:), allocatable **pot\_fr**  
*fraction of HRU area that drains into pothole ( $\text{km}^2/\text{km}^2$ )*
- real \*8, dimension(:), allocatable **pot\_vol**  
*initial or current volume of water stored in the depression/impounded area (read in as mm and converted to  $\text{m}^3$ )*  
*(needed only if current HRU is IPOT) (mm or  $\text{m}^3 \text{H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable **potsa**  
*surface area of impounded water body (ha)*
- real \*8, dimension(:), allocatable **wfsh**  
*wetting front matric potential (average capillary suction at wetting front) (mm)*
- real \*8, dimension(:), allocatable **potflwi**  
*water entering pothole on day ( $\text{m}^3 \text{H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable **potsedi**  
*sediment entering pothole on day (metric tons)*
- real \*8, dimension(:), allocatable **newrti**  
*infiltration rate for last time step from the previous day (mm/hr)*
- real \*8, dimension(:), allocatable **fsred**  
*reduction in bacteria loading from filter strip (none)*
- real \*8, dimension(:), allocatable **pot\_no3**  
*amount of nitrate in pothole water body (kg N)*
- real \*8, dimension(:), allocatable **pot\_sed**  
*amount of sediment in pothole water body (metric tons)*
- real \*8, dimension(:), allocatable **dis\_stream**  
*average distance to stream (m)*
- real \*8, dimension(:), allocatable **sed\_con**
- real \*8, dimension(:), allocatable **orgn\_con**
- real \*8, dimension(:), allocatable **orgp\_con**
- real \*8, dimension(:), allocatable **pot\_k**  
*hydraulic conductivity of soil surface of pothole defaults to conductivity of upper soil (0.↵  
01-10.) layer*
- real \*8, dimension(:), allocatable **soln\_con**

- real \*8, dimension(:), allocatable **solp\_con**
- real \*8, dimension(:), allocatable **n\_reduc**  
*nitrogen uptake reduction factor (not currently used; defaulted 300.)*
- real \*8, dimension(:), allocatable **n\_ln**  
*power function exponent for calculating nitrate concentration in subsurface drains (1.0 - 3.0) (dimensionless)*
- real \*8, dimension(:), allocatable **n\_lnc**  
*coefficient for power function for calculating nitrate concentration in subsurface drains (0.5 - 4.0) (dimensionless)*
- integer, dimension(:), allocatable **ioper**
- real \*8, dimension(:), allocatable **usle\_ls**  
*USLE equation length slope (LS) factor (none)*
- real \*8, dimension(:), allocatable **filterw**  
*filter strip width for bacteria transport (m)*
- real \*8, dimension(:), allocatable **phuacc**  
*fraction of plant heat units accumulated (none)*
- real \*8, dimension(:), allocatable **sumix**  
*sum of all tillage mixing efficiencies for HRU operation (none)*
- real \*8, dimension(:), allocatable **epco**  
*plant water uptake compensation factor (0-1) (none)*
- real \*8, dimension(:), allocatable **esco**  
*soil evaporation compensation factor (0-1) (none)*
- real \*8, dimension(:), allocatable **hru\_slp**  
*average slope steepness in HRU (m/m)*
- real \*8, dimension(:), allocatable **slsubbsn**  
*average slope length for subbasin (m)*
- real \*8, dimension(:), allocatable **erorgn**  
*organic N enrichment ratio, if left blank the model will calculate for every event (none)*
- real \*8, dimension(:), allocatable **erorgp**  
*organic P enrichment ratio, if left blank the model will calculate for every event (none)*
- real \*8, dimension(:), allocatable **biomix**  
*biological mixing efficiency. Mixing of soil due to activity of earthworms and other soil biota. Mixing is performed at the end of every calendar year (none)*
- real \*8, dimension(:), allocatable **pnd\_seci**  
*secchi-disk depth of pond (m)*
- real \*8, dimension(:), allocatable **canmx**  
*maximum canopy storage (mm H2O)*
- real \*8, dimension(:), allocatable **divmax**  
*maximum daily irrigation diversion from the reach (when IRRSC=1 or IRR=3): when value is positive the units are mm H2O; when the value is negative, the units are (10<sup>4</sup> m<sup>3</sup> H2O) (mm H2O or 10<sup>4</sup> m<sup>3</sup> H2O)*
- real \*8, dimension(:), allocatable **flowmin**  
*minimum instream flow for irrigation diversions when IRRSC=1, irrigation water will be diverted only when streamflow is at or above FLOWMIN (m<sup>3</sup>/s)*
- real \*8, dimension(:), allocatable **usle\_p**  
*USLE equation support practice (P) factor (none)*
- real \*8, dimension(:), allocatable **lat\_sed**  
*sediment concentration in lateral flow (g/L)*
- real \*8, dimension(:), allocatable **rch\_dakm**  
*total drainage area contributing to flow at the outlet (pour point) of the reach in square kilometers (km<sup>2</sup>)*
- real \*8, dimension(:, :), allocatable **cn**  
*cn(1,:) SCS runoff curve number for moisture condition I (none)*  
*cn(2,:) SCS runoff curve number for moisture condition II (none)*  
*cn(3,:) SCS runoff curve number for moisture condition III (none)*
- real \*8, dimension(:), allocatable **pnd\_no3s**

- amount of nitrate originating from lateral flow in pond at end of day or at beginning of day (kg N)*

  - real \*8, dimension(:), allocatable [lat\\_ttime](#)
- lateral flow travel time or exponential of the lateral flow travel time (days or none)*

  - real \*8, dimension(:), allocatable [flowfr](#)
- fraction of available flow in reach that is allowed to be applied to the HRU (none)*

  - real \*8, dimension(:), allocatable [sol\\_zmx](#)
- maximum rooting depth (mm)*

  - real \*8, dimension(:), allocatable [tile\\_ttime](#)
- exponential of the tile flow travel time (none)*

  - real \*8, dimension(:), allocatable [slsoil](#)
- slope length for lateral subsurface flow (m)*

  - real \*8, dimension(:), allocatable [gwmimp](#)
- soluble P concentration in groundwater loading to reach (mg P/L)*

  - real \*8, dimension(:), allocatable [sol\\_cov](#)
- amount of residue on soil surface (kg/ha)*

  - real \*8, dimension(:), allocatable [sed\\_stl](#)
- fraction of sediment remaining suspended in impoundment after settling for one day (kg/kg)*

  - real \*8, dimension(:), allocatable [ov\\_n](#)
- Manning's "n" value for overland flow (none)*

  - real \*8, dimension(:), allocatable [pnd\\_no3](#)
- amount of nitrate originating from surface runoff in pond at end of day or at beginning of day (kg N)*

  - real \*8, dimension(:), allocatable [pnd\\_solp](#)
- amount of soluble P originating from surface runoff in pond at end of day or at beginning of day (kg P)*

  - real \*8, dimension(:), allocatable [yldanu](#)
- annual yield (dry weight) in the HRU (metric tons/ha)*

  - real \*8, dimension(:), allocatable [pnd\\_orgn](#)
- amount of organic N originating from surface runoff in pond at end of day or at beginning of day (kg N)*

  - real \*8, dimension(:), allocatable [pnd\\_organ](#)
- amount of organic P originating from surface runoff in pond at end of day or at beginning of day (kg P)*

  - real \*8, dimension(:), allocatable [twlpnd](#)
- water lost through seepage from ponds on day in HRU (mm H2O)*

  - real \*8, dimension(:), allocatable [twlwet](#)
- water lost through seepage from wetlands on day in HRU (mm H2O)*

  - real \*8, dimension(:), allocatable [hru\\_fr](#)
- fraction of subbasin area contained in HRU (km<sup>2</sup>/km<sup>2</sup>)*

  - real \*8, dimension(:), allocatable [sol\\_sumul](#)
- amount of water held in soil profile at saturation (mm H2O)*

  - real \*8, dimension(:), allocatable [pnd\\_chla](#)
- amount of chlorophyll-a in pond at end of day (kg chl\_a)*

  - real \*8, dimension(:), allocatable [hru\\_km](#)
- area of HRU in square kilometers (km<sup>2</sup>)*

  - real \*8, dimension(:), allocatable [bio\\_ms](#)
- land cover/crop biomass (dry weight) (kg/ha)*

  - real \*8, dimension(:), allocatable [sol\\_alb](#)
- albedo when soil is moist (none)*

  - real \*8, dimension(:), allocatable [strsw](#)
- fraction of potential plant growth achieved on the day where the reduction is caused by water stress (none)*

  - real \*8, dimension(:), allocatable [pnd\\_fr](#)
- fraction of HRU/subbasin area that drains into ponds (none)*

  - real \*8, dimension(:), allocatable [pnd\\_k](#)
- hydraulic conductivity through bottom of ponds (mm/hr)*

- real \*8, dimension(:), allocatable [pnd\\_psa](#)  
*surface area of ponds when filled to principal spillway (ha)*
- real \*8, dimension(:), allocatable [pnd\\_pvol](#)  
*runoff volume of water from catchment area needed to fill the ponds to the principal spillway (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [pnd\\_esa](#)  
*surface area of ponds when filled to emergency spillway (ha)*
- real \*8, dimension(:), allocatable [pnd\\_evol](#)  
*runoff volume of water from catchment area needed to fill the ponds to the emergency spillway (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [pnd\\_vol](#)  
*volume of water in ponds (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [yldaa](#)  
*average annual yield (dry weight) in the HRU (metric tons)*
- real \*8, dimension(:), allocatable [pnd\\_nsed](#)  
*normal sediment concentration in pond water (UNIT CHANGE!) (mg/kg or kg/kg)*
- real \*8, dimension(:), allocatable [pnd\\_sed](#)  
*sediment concentration in pond water (UNIT CHANGE!) (mg/kg or kg/kg)*
- real \*8, dimension(:), allocatable [dep\\_imp](#)  
*depth to impervious layer (mm)*
- real \*8, dimension(:), allocatable **strsa**
- real \*8, dimension(:), allocatable **evpnd**
- real \*8, dimension(:), allocatable **evwet**
- real \*8, dimension(:), allocatable [wet\\_fr](#)  
*fraction of HRU/subbasin area that drains into wetlands (none)*
- real \*8, dimension(:), allocatable [wet\\_k](#)  
*hydraulic conductivity of bottom of wetlands (mm/hr)*
- real \*8, dimension(:), allocatable [wet\\_nsa](#)  
*surface area of wetlands in subbasin at normal water level (ha)*
- real \*8, dimension(:), allocatable [wet\\_nvol](#)  
*runoff volume of water from catchment area needed to fill wetlands to normal water level (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- integer, dimension(:), allocatable **iwetgw**
- integer, dimension(:), allocatable **iwetile**
- real \*8, dimension(:), allocatable [wet\\_mxsa](#)  
*surface area of wetlands at maximum water level (ha)*
- real \*8, dimension(:), allocatable [wet\\_mxvol](#)  
*runoff volume of water from catchment area needed to fill wetlands to maximum water level (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [wet\\_vol](#)  
*volume of water in wetlands (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [wet\\_nsed](#)  
*normal sediment concentration in wetland water (UNIT CHANGE!) (mg/kg or kg/kg)*
- real \*8, dimension(:), allocatable [wet\\_sed](#)  
*sediment concentration in wetland water (UNIT CHANGE!) (mg/L or kg/L)*
- real \*8, dimension(:,:), allocatable [bp](#)  
*bp(1,:) 1st shape parameter for the pond surface area equation (none)*  
*bp(2,:) 2nd shape parameter for the pond surface area equation (none)*
- real \*8, dimension(:), allocatable [sci](#)  
*retention coefficient for CN method based on plant ET (none)*
- real \*8, dimension(:), allocatable [smx](#)  
*retention coefficient for CN method based on soil moisture (none)*

- real \*8, dimension(:,:), allocatable **bw**  
*bw(1,:) 1st shape parameter for the wetland surface area equation (none)*  
*bw(2,:) 2nd shape parameter for the wetland surface area equation (none)*
- real \*8, dimension(:), allocatable **bactpq**  
*persistent bacteria in soil solution (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **cnday**  
*curve number for current day, HRU and at current soil moisture (none)*
- real \*8, dimension(:), allocatable **bactlp\_plt**  
*less persistent bacteria on foliage (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **bactp\_plt**  
*persistent bacteria on foliage (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **auto\_eff**  
*fertilizer application efficiency calculated as the amount of N applied divided by the amount of N removed at harvest (none)*
- real \*8, dimension(:), allocatable **secciw**  
*water clarity coefficient for wetland (none)*
- real \*8, dimension(:), allocatable **sol\_sw**  
*amount of water stored in soil profile at end of any given day (mm H<sub>2</sub>O)*
- real \*8, dimension(:), allocatable **bactlpq**  
*less persistent bacteria in soil solution (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **chlawn**  
*chlorophyll-a production coefficient for wetland (none)*
- real \*8, dimension(:), allocatable **tmpav**  
*average air temperature on current day in HRU (deg C)*
- real \*8, dimension(:), allocatable **bactlps**  
*less persistent bacteria attached to soil particles (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **bactps**  
*persistent bacteria attached to soil particles (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **sno\_hru**  
*amount of water stored as snow in HRU on current day (mm H<sub>2</sub>O)*
- real \*8, dimension(:), allocatable **wet\_orgn**  
*amount of organic N originating from surface runoff in wetland at end of day (kg N)*
- real \*8, dimension(:), allocatable **hru\_ra**  
*solar radiation for the day in HRU (MJ/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **subp**  
*precipitation for the day in HRU (mm H<sub>2</sub>O)*
- real \*8, dimension(:), allocatable **rsdin**  
*initial residue cover (kg/ha)*
- real \*8, dimension(:), allocatable **tmn**  
*minimum air temperature on current day in HRU (deg C)*
- real \*8, dimension(:), allocatable **tmx**  
*maximum air temperature on current day in HRU (deg C)*
- real \*8, dimension(:), allocatable **tmp\_hi**  
*last maximum temperature in HRU (deg C)*
- real \*8, dimension(:), allocatable **tmp\_lo**  
*last minimum temperature in HRU (deg C)*
- real \*8, dimension(:), allocatable **usle\_k**  
*USLE equation soil erodibility (K) factor (none)*
- real \*8, dimension(:), allocatable **tconc**  
*time of concentration for HRU (hour)*
- real \*8, dimension(:), allocatable **hru\_rmx**

- maximum possible solar radiation for the day in HRU (MJ/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable [rwt](#)
- fraction of total plant biomass that is in roots (none)*
- real \*8, dimension(:), allocatable [olai](#)
- real \*8, dimension(:), allocatable [usle\\_cfac](#)
- real \*8, dimension(:), allocatable [usle\\_eifac](#)
- real \*8, dimension(:), allocatable [sol\\_sumfc](#)
- amount of water held in soil profile at field capacity (mm H2O)*
- real \*8, dimension(:), allocatable [t\\_ov](#)
- time for flow from farthest point in subbasin to enter a channel (hour)*
- real \*8, dimension(:), allocatable [anano3](#)
- total amount of NO3 applied during the year in auto-fertilization (kg N/ha)*
- real \*8, dimension(:), allocatable [aird](#)
- amount of water applied to HRU on current day (mm H2O)*
- real \*8, dimension(:), allocatable [wet\\_orgp](#)
- amount of organic P originating from surface runoff in wetland at end of day (kg P)*
- real \*8, dimension(:), allocatable [usle\\_mult](#)
- product of USLE K,P,LS,exp(rock) (none)*
- real \*8, dimension(:), allocatable [rhd](#)
- relative humidity for the day in HRU (none)*
- real \*8, dimension(:), allocatable [u10](#)
- wind speed (measured at 10 meters above surface) for the day in HRU (m/s)*
- real \*8, dimension(:), allocatable [cht](#)
- canopy height (m)*
- real \*8, dimension(:), allocatable [aairr](#)
- average annual amount of irrigation water applied to HRU (mm H2O)*
- real \*8, dimension(:), allocatable [lai\\_aamx](#)
- maximum leaf area index for the entire period of simulation in the HRU (none)*
- real \*8, dimension(:), allocatable [deepirr](#)
- amount of water removed from deep aquifer for irrigation (mm H2O)*
- real \*8, dimension(:), allocatable [shallirr](#)
- amount of water removed from shallow aquifer for irrigation (mm H2O)*
- real \*8, dimension(:), allocatable [wet\\_no3](#)
- amount of nitrate originating from surface runoff in wetland at end of day (kg N)*
- real \*8, dimension(:), allocatable [ovrlnd](#)
- overland flow onto HRU from upstream routing unit (mm H2O)*
- real \*8, dimension(:), allocatable [canstor](#)
- amount of water held in canopy storage (mm H2O)*
- real \*8, dimension(:), allocatable [irr\\_mx](#)
- maximum irrigation amount per auto application (mm)*
- real \*8, dimension(:), allocatable [auto\\_wstr](#)
- water stress factor which triggers auto irrigation (none or mm)*
- integer, dimension(:), allocatable [cfrt\\_id](#)
- fertilizer/manure identification number from database (fert.dat) (none)*
- real \*8, dimension(:), allocatable [cfrt\\_kg](#)
- amount of fertilizer/manure applied to HRU on a given day ((kg/ha)/day)*
- integer, dimension(:), allocatable [cpst\\_id](#)
- real \*8, dimension(:), allocatable [cpst\\_kg](#)
- real \*8, dimension(:), allocatable [irr\\_asq](#)
- surface runoff ratio*
- real \*8, dimension(:), allocatable [irr\\_eff](#)



- real \*8, dimension(:), allocatable **irrsq**  
*surface runoff ratio (0-1) .1 is 10% surface runoff (frac)*
- real \*8, dimension(:), allocatable **irrefm**
- real \*8, dimension(:), allocatable **bio\_eat**  
*dry weight of biomass removed by grazing daily ((kg/ha)/day)*
- real \*8, dimension(:), allocatable **bio\_trmp**  
*dry weight of biomass removed by trampling daily ((kg/ha)/day)*
- integer, dimension(:), allocatable **ipst\_freq**  
*number of days between applications (days)*
- integer, dimension(:), allocatable **ifrt\_freq**  
*number of days between applications in continuous fertilizer operation (days)*
- integer, dimension(:), allocatable **irr\_noa**
- integer, dimension(:), allocatable **irr\_sc**
- integer, dimension(:), allocatable **irr\_no**
- integer, dimension(:), allocatable **imp\_trig**  
*release/impound action code (none):*  
*0 begin impounding water*  
*1 release impounded water*
- integer, dimension(:), allocatable **fert\_days**  
*number of days continuous fertilization will be simulated (none)*
- integer, dimension(:), allocatable **irr\_sca**
- integer, dimension(:), allocatable **idplt**  
*land cover/crop identification code for first crop grown in HRU (the only crop if there is no rotation) (from crop.dat)*  
*(none)*
- integer, dimension(:), allocatable **wstrs\_id**  
*water stress identifier (none):*  
*1 plant water demand*  
*2 soil water deficit*
- integer, dimension(:), allocatable **pest\_days**
- real \*8, dimension(:,:), allocatable **bio\_aahv**  
*harvested biomass of plant (kg/ha)*
- real \*8, dimension(:), allocatable **wet\_solp**  
*amount of soluble P originating from surface runoff in wetland at end of day (kg P)*
- real \*8, dimension(:), allocatable **wet\_chla**  
*amount of chlorophyll-a in wetland at end of day (kg chla)*
- real \*8, dimension(:), allocatable **wet\_no3s**  
*amount of nitrate originating from lateral flow in wetland at end of day (kg N)*
- real \*8, dimension(:), allocatable **pstsol**  
*amount of soluble pesticide leached from bottom of soil profile on current day (kg pst/ha)*
- real \*8, dimension(:), allocatable **pnd\_no3g**  
*amount of nitrate originating from groundwater in pond at end of day or at beginning of day (kg N)*
- real \*8, dimension(:), allocatable **wet\_seci**  
*secchi-disk depth in wetland at end of day (m)*
- real \*8, dimension(:), allocatable **delay**  
*groundwater delay: time required for water leaving the bottom of the root zone to reach the shallow aquifer (days)*
- real \*8, dimension(:), allocatable **gwht**  
*groundwater height (m)*
- real \*8, dimension(:), allocatable **gw\_q**  
*groundwater contribution to streamflow from HRU on current day (mm H2O)*
- real \*8, dimension(:), allocatable **pnd\_solpg**  
*amount of soluble P originating from groundwater in pond at end of day or at beginning of day (kg P)*
- real \*8, dimension(:), allocatable **alpha\_bf**



- alpha factor for groundwater recession curve (1/days)*

  - real \*8, dimension(:), allocatable [alpha\\_bfe](#)  
 *$\exp(-\alpha_{bf}f)$  (none)*
- specific yield for shallow aquifer ( $m^3/m^3$ )*

  - real \*8, dimension(:), allocatable [gw\\_spyld](#)
- alpha factor for groundwater recession curve of the deep aquifer (1/days)*

  - real \*8, dimension(:), allocatable [alpha\\_bfe\\_d](#)  
 *$\exp(-\alpha_{bf_d}f_d)$  (with  $\alpha_{bf_d}$  the alpha factor for groundwater recession curve of the deep aquifer (1/days)) (none)*
- groundwater contribution to streamflow from deep aquifer from HRU on current day (mm H2O)*

  - real \*8, dimension(:), allocatable [gw\\_qdeep](#)
- groundwater delay (time required for water leaving the bottom of the root zone to reach the shallow aquifer; units-days) (none)*

  - real \*8, dimension(:), allocatable [gw\\_delaye](#)  
 *$\exp(-1/delay)$  where  $delay(:)$  is the groundwater delay (time required for water leaving the bottom of the root zone to reach the shallow aquifer; units-days) (none)*
- revap coeff: this variable controls the amount of water moving from the shallow aquifer to the root zone as a result of soil moisture depletion (none)*

  - real \*8, dimension(:), allocatable [gw\\_revap](#)
- recharge to deep aquifer: the fraction of root zone percolation that reaches the deep aquifer (none)*

  - real \*8, dimension(:), allocatable [rchrg\\_dp](#)
- fraction of porosity from which anions are excluded*

  - real \*8, dimension(:), allocatable [anion\\_excl](#)
- threshold depth of water in shallow aquifer required to allow revap to occur (mm H2O)*

  - real \*8, dimension(:), allocatable [revapmn](#)
- amount of water recharging both aquifers on current day in HRU (mm H2O)*

  - real \*8, dimension(:), allocatable [rchrg](#)
- minimum plant biomass for grazing (kg/ha)*

  - real \*8, dimension(:), allocatable [bio\\_min](#)
- initial HRU soil water content expressed as fraction of field capacity (none)*

  - real \*8, dimension(:), allocatable [ffc](#)
- amount of soluble phosphorus in surface runoff in HRU for the day (kg P/ha)*

  - real \*8, dimension(:), allocatable [surqsolp](#)
- depth of water in deep aquifer (mm H2O)*

  - real \*8, dimension(:), allocatable [deepst](#)
- depth of water in shallow aquifer in HRU (mm H2O)*

  - real \*8, dimension(:), allocatable [shallst](#)
- amount of soluble P originating from groundwater in wetland at end of day (kg P)*

  - real \*8, dimension(:), allocatable [wet\\_solpg](#)
- filter strip trapping efficiency (used for everything but bacteria) (none)*

  - real \*8, dimension(:), allocatable [cklsp](#)
- filter strip trapping efficiency (used for everything but bacteria) (none)*

  - real \*8, dimension(:), allocatable [rchrg\\_src](#)
- filter strip trapping efficiency (used for everything but bacteria) (none)*

  - real \*8, dimension(:), allocatable [trapeff](#)
- average bulk density for soil profile ( $Mg/m^3$ )*

  - real \*8, dimension(:), allocatable [sol\\_avbd](#)
- amount of nitrate originating from groundwater in wetland at end of day (kg N)*

  - real \*8, dimension(:), allocatable [wet\\_no3g](#)
- time to drain soil to field capacity yield used in autofertilization (hours)*

  - real \*8, dimension(:), allocatable [tdrain](#)
- threshold depth of water in shallow aquifer required before groundwater flow will occur (mm H2O)*

  - real \*8, dimension(:), allocatable [gwqmn](#)
- temperature of snow pack in HRU (deg C)*

  - real \*8, dimension(:), allocatable [snotmp](#)
- temperature of snow pack in HRU (deg C)*

  - real \*8, dimension(:), allocatable [pplnt](#)

- plant uptake of phosphorus in HRU for the day (kg P/ha)*

  - real \*8, dimension(:), allocatable [gdrain](#)
- drain tile lag time: the amount of time between the transfer of water from the soil to the drain tile and the release of the water from the drain tile to the reach (hours)*

  - real \*8, dimension(:), allocatable [ddrain](#)
- depth of drain tube from the soil surface (mm)*

  - real \*8, dimension(:), allocatable [sol\\_crk](#)
- crack volume potential of soil (none)*

  - real \*8, dimension(:), allocatable [brt](#)
- fraction of surface runoff within the subbasin which takes 1 day or less to reach the subbasin outlet (none)*

  - real \*8, dimension(:), allocatable [dayl](#)
- length of the current day in HRU (hours)*

  - real \*8, dimension(:), allocatable [sstmaxd](#)
- static maximum depressional storage; read from .sdr (mm)*

  - real \*8, dimension(:), allocatable [re](#)
- effective radius of drains (mm)*

  - real \*8, dimension(:), allocatable [sdrain](#)
- distance between two drain tubes or tiles (mm)*

  - real \*8, dimension(:), allocatable [drain\\_co](#)
- drainage coefficient (mm/day)*

  - real \*8, dimension(:), allocatable [latksatf](#)
- multiplication factor to determine  $conk(j1,j)$  from  $sol\_k(j1,j)$  for HRU (none)*

  - real \*8, dimension(:), allocatable [pc](#)
- pump capacity (default pump capacity = 1.042mm/hr or 25mm/day) (mm/hr)*

  - real \*8, dimension(:), allocatable [stmaxd](#)
- maximum surface depressional storage for day in a given HRU (mm)*

  - real \*8, dimension(:), allocatable [rnd3](#)
- random number between 0.0 and 1.0 (none)*

  - real \*8, dimension(:), allocatable [rnd2](#)
- random number between 0.0 and 1.0 (none)*

  - real \*8, dimension(:), allocatable [twash](#)
- time that solids have built-up on streets (days)*

  - real \*8, dimension(:), allocatable [doxq](#)
- dissolved oxygen concentration in the surface runoff on current day in HRU (mg/L)*

  - real \*8, dimension(:), allocatable [sol\\_cnsw](#)
- amount of water stored in soil profile used to calculate daily CN value (initial soil water content for day) (mm H2O)*

  - real \*8, dimension(:), allocatable [rnd8](#)
- random number between 0.0 and 1.0 (none)*

  - real \*8, dimension(:), allocatable [rnd9](#)
- random number between 0.0 and 1.0 (none)*

  - real \*8, dimension(:), allocatable [percn](#)
- amount of nitrate percolating past bottom of soil profile during the day (kg N/ha)*

  - real \*8, dimension(:), allocatable [sol\\_sumwp](#)
- total or net amount of water entering main channel for day from HRU (mm H2O)*

  - real \*8, dimension(:), allocatable [qdr](#)
- amount of N applied in autofert operation in year (kg N/ha) (mg/L)*

  - real \*8, dimension(:), allocatable [chl\\_a](#)
- chlorophyll-a concentration in water yield on current day in HRU (microgram/L)*

  - real \*8, dimension(:), allocatable [latq](#)
- total amount of water in lateral flow in soil profile for the day in HRU (mm H2O)*

- real \*8, dimension(:), allocatable [nplnt](#)  
*plant uptake of nitrogen in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [latno3](#)  
*amount of nitrate transported with lateral flow in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [minpgw](#)  
*soluble P loading to reach in groundwater (kg P/ha)*
- real \*8, dimension(:), allocatable [no3gw](#)  
*nitrate loading to reach in groundwater (kg N/ha)*
- real \*8, dimension(:), allocatable [tileq](#)
- real \*8, dimension(:), allocatable [tileno3](#)
- real \*8, dimension(:), allocatable [sedorgn](#)  
*amount of organic nitrogen in surface runoff in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [sedminpa](#)  
*amount of active mineral phosphorus sorbed to sediment in surface runoff in HRU for day (kg P/ha)*
- real \*8, dimension(:), allocatable [sedminps](#)  
*amount of stable mineral phosphorus sorbed to sediment in surface runoff in HRU for day (kg P/ha)*
- real \*8, dimension(:), allocatable [sedyld](#)  
*soil loss caused by water erosion for day in HRU (metric tons)*
- real \*8, dimension(:), allocatable [sepbtm](#)  
*percolation from bottom of soil profile for the day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [strsn](#)  
*fraction of potential plant growth achieved on the day where the reduction is caused by nitrogen stress (none)*
- real \*8, dimension(:), allocatable [sedorgp](#)  
*amount of organic phosphorus in surface runoff in HRU for the day (kg P/ha)*
- real \*8, dimension(:), allocatable [surfq](#)  
*surface runoff generated in HRU on the current day (mm H2O)*
- real \*8, dimension(:), allocatable [strstmp](#)  
*fraction of potential plant growth achieved on the day in HRU where the reduction is caused by temperature stress (none)*
- real \*8, dimension(:), allocatable [surqno3](#)  
*amount of nitrate transported in surface runoff in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [hru\\_ha](#)  
*area of HRU in hectares (ha)*
- real \*8, dimension(:), allocatable [hru\\_dafr](#)  
*fraction of total watershed area contained in HRU (km2/km2)*
- real \*8, dimension(:), allocatable [drydep\\_no3](#)  
*atmospheric dry deposition of nitrates (kg/ha/yr)*
- real \*8, dimension(:), allocatable [drydep\\_nh4](#)  
*atmospheric dry deposition of ammonia (kg/ha/yr)*
- real \*8, dimension(:), allocatable [bio\\_yrms](#)  
*annual biomass (dry weight) in the HRU (metric tons/ha)*
- real \*8, dimension(:), allocatable [phubase](#)  
*base zero total heat units (used when no land cover is growing) (heat units)*
- real \*8, dimension(:), allocatable [hvstiadj](#)  
*optimal harvest index adjusted for water stress for current time during growing season ((kg/ha)/(kg/ha))*
- real \*8, dimension(:), allocatable [laiday](#)  
*leaf area index for HRU ( $m^2/m^2$ )*
- real \*8, dimension(:), allocatable [chlap](#)  
*chlorophyll-a production coefficient for pond (none)*
- real \*8, dimension(:), allocatable [pnd\\_psed](#)

- amount of mineral P attached to sediment originating from surface runoff in pond at end of day or beginning of day (kg P)*
- real \*8, dimension(:), allocatable **laimxfr**
- real \*8, dimension(:), allocatable **seccip**
- water clarity coefficient for pond (none)*
- real \*8, dimension(:), allocatable **plantn**
- amount of nitrogen in plant biomass (kg N/ha)*
- real \*8, dimension(:), allocatable **plt\_et**
- actual ET simulated during life of plant (mm H2O)*
- real \*8, dimension(:), allocatable **wet\_psed**
- amount of mineral P attached to sediment originating from surface runoff in wetland at end of day (kg P)*
- real \*8, dimension(:), allocatable **bio\_aams**
- average annual biomass (dry weight) in the HRU (metric tons)*
- real \*8, dimension(:), allocatable **plantp**
- amount of phosphorus stored in plant biomass (kg P/ha)*
- real \*8, dimension(:), allocatable **plt\_pet**
- potential ET simulated during life of plant (mm H2O)*
- real \*8, dimension(:), allocatable **dormhr**
- time threshold used to define dormant period for plant (when daylength is within the time specified by dl from the minimum daylength for the area, the plant will go dormant) (hour)*
- real \*8, dimension(:), allocatable **lai\_yrmx**
- maximum leaf area index for the year in the HRU (none)*
- real \*8, dimension(:), allocatable **lat\_pst**
- amount of pesticide in lateral flow in HRU for the day (kg pst/ha)*
- real \*8, dimension(:), allocatable **orig\_snohru**
- real \*8, dimension(:), allocatable **orig\_potvol**
- real \*8, dimension(:), allocatable **pltfr\_n**
- fraction of plant biomass that is nitrogen (none)*
- real \*8, dimension(:), allocatable **orig\_alai**
- real \*8, dimension(:), allocatable **orig\_bioms**
- real \*8, dimension(:), allocatable **pltfr\_p**
- fraction of plant biomass that is phosphorus (none)*
- real \*8, dimension(:), allocatable **orig\_phuacc**
- real \*8, dimension(:), allocatable **orig\_sumix**
- real \*8, dimension(:), allocatable **phu\_plt**
- total number of heat units to bring plant to maturity (heat units)*
- real \*8, dimension(:), allocatable **orig\_phu**
- real \*8, dimension(:), allocatable **orig\_shallst**
- real \*8, dimension(:), allocatable **orig\_deepst**
- real \*8, dimension(:), allocatable **orig\_pndvol**
- real \*8, dimension(:), allocatable **orig\_pndsed**
- real \*8, dimension(:), allocatable **orig\_pndno3**
- real \*8, dimension(:), allocatable **orig\_pndsolp**
- real \*8, dimension(:), allocatable **orig\_pndorgn**
- real \*8, dimension(:), allocatable **orig\_pndorgp**
- real \*8, dimension(:), allocatable **orig\_wetvol**
- real \*8, dimension(:), allocatable **orig\_wetsed**
- real \*8, dimension(:), allocatable **orig\_wetno3**
- real \*8, dimension(:), allocatable **orig\_wetsolp**
- real \*8, dimension(:), allocatable **orig\_wetorgn**
- real \*8, dimension(:), allocatable **orig\_wetorgp**
- real \*8, dimension(:), allocatable **orig\_solcov**
- real \*8, dimension(:), allocatable **orig\_solsw**

- real \*8, dimension(:), allocatable **orig\_potno3**
- real \*8, dimension(:), allocatable **orig\_potsed**
- real \*8, dimension(:), allocatable **wtab**  
*water table based on 30 day antecedent climate (precip,et) (mm)*
- real \*8, dimension(:), allocatable **shallst\_n**  
*nitrate concentration in shallow aquifer converted to kg/ha (ppm NO3-N)*
- real \*8, dimension(:), allocatable **gw\_nloss**
- real \*8, dimension(:), allocatable **rchrg\_n**
- real \*8, dimension(:), allocatable **det\_san**
- real \*8, dimension(:), allocatable **det\_sil**
- real \*8, dimension(:), allocatable **det\_cla**
- real \*8, dimension(:), allocatable **det\_sag**
- real \*8, dimension(:), allocatable **det\_lag**
- real \*8, dimension(:), allocatable **afrt\_surface**  
*fraction of fertilizer which is applied to top 10 mm of soil (the remaining fraction is applied to first soil layer) (none)*
- real \*8, dimension(:), allocatable **tnylda**  
*estimated/target nitrogen content of yield used in autofertilization (kg N/kg yield)*
- real \*8 **frt\_surface**  
*fraction of fertilizer which is applied to the top 10 mm of soil (the remaining fraction is applied to the first soil layer) (none)*
- real \*8, dimension(:), allocatable **auto\_nyr**  
*maximum NO3-N content allowed to be applied in one year by auto-fertilization (kg NO3-N/ha)*
- real \*8, dimension(:), allocatable **auto\_napp**  
*maximum NO3-N content allowed in one fertilizer application (kg NO3-N/ha)*
- real \*8, dimension(:), allocatable **auto\_nstrs**  
*nitrogen stress factor which triggers auto fertilization (none)*
- real \*8, dimension(:), allocatable **manure\_kg**  
*dry weight of manure deposited on HRU daily ((kg/ha)/day)*
- real \*8, dimension(:,:), allocatable **rcn\_mo**
- real \*8, dimension(:,:), allocatable **rammo\_mo**
- real \*8, dimension(:,:), allocatable **drydep\_no3\_mo**
- real \*8, dimension(:,:), allocatable **drydep\_nh4\_mo**
- real \*8, dimension(:), allocatable **rcn\_d**
- real \*8, dimension(:), allocatable **rammo\_d**
- real \*8, dimension(:), allocatable **drydep\_no3\_d**
- real \*8, dimension(:), allocatable **drydep\_nh4\_d**
- real \*8, dimension(:,:), allocatable **yldn**  
*average value for yield of crop (kg/ha)*
- integer, dimension(:,:), allocatable **gwati**
- real \*8, dimension(:,:), allocatable **gwatn**
- real \*8, dimension(:,:), allocatable **gwatl**
- real \*8, dimension(:,:), allocatable **gwatw**
- real \*8, dimension(:,:), allocatable **gwatd**
- real \*8, dimension(:,:), allocatable **gwats**
- real \*8, dimension(:,:), allocatable **gwatspcon**
- real \*8, dimension(:,:), allocatable **psetlp**  
*psetlp(1,:) phosphorus settling rate for 1st season (m/day)*  
*psetlp(2,:) phosphorus settling rate for 2nd season (m/day)*
- real \*8, dimension(:,:), allocatable **wgnold**  
*previous value of wgncur(:,:) (none)*
- real \*8, dimension(:,:), allocatable **wgncur**

- parameter to predict the impact of precip on other weather attributes (none)*
- wgncur(1,:) parameter which predicts impact of precip on daily maximum air temperature*
- wgncur(2,:) parameter which predicts impact of precip on daily minimum air temperature*
- wgncur(3,:) parameter which predicts impact of precip on daily solar radiation*
- real \*8, dimension(:,:), allocatable [wrt](#)
  - wrt(1,:) 1st shape parameter for calculation of water retention (none)*
  - wrt(2,:) 2nd shape parameter for calculation of water retention (none)*
- real \*8, dimension(:,:), allocatable [pst\\_enr](#)
  - pesticide enrichment ratio (none)*
- real \*8, dimension(:,:), allocatable [pst\\_surq](#)
  - amount of pesticide type lost in water surface runoff on current day in HRU (kg/ha)*
- real \*8, dimension(:,:), allocatable [zdb](#)
  - division term from net pesticide equation (mm)*
- real \*8, dimension(:,:), allocatable [plt\\_pst](#)
  - pesticide on plant foliage (kg/ha)*
- real \*8, dimension(:,:), allocatable [psetlw](#)
  - psetlw(1,:) phosphorus settling rate for 1st season (m/day)*
  - psetlw(2,:) phosphorus settling rate for 2nd season (m/day)*
- real \*8, dimension(:,:), allocatable [pst\\_sed](#)
  - pesticide loading from HRU sorbed onto sediment (kg/ha)*
- real \*8, dimension(:,:), allocatable [wupnd](#)
  - average daily water removal from the pond for the month for the HRU within the subbasin ( $10^4 \text{ m}^3/\text{day}$ )*
- real \*8, dimension(:,:), allocatable [phi](#)
  - phi(1,:) cross-sectional area of flow at bankfull depth ( $\text{m}^2$ ) phi(2,:) (none) phi(3,:) (none) phi(4,:) (none) phi(5,:) flow rate when reach is at bankfull depth ( $\text{m}^3/\text{s}$ ) phi(6,:) bottom width of main channel (m) phi(7,:) depth of water when reach is at bankfull depth (m) phi(8,:) average velocity when reach is at bankfull depth (m/s) phi(9,:) wave celerity when reach is at bankfull depth (m/s) phi(10,:) storage time constant for reach at bankfull depth (ratio of storage to discharge) (hour) phi(11,:) average velocity when reach is at 0.1 bankfull depth (low flow) (m/s) phi(12,:) wave celerity when reach is at 0.1 bankfull depth (low flow) (m/s) phi(13,:) storage time constant for reach at 0.1 bankfull depth (low flow) (ratio of storage to discharge) (hour)*
- real \*8, dimension(:,:), allocatable [pcpband](#)
  - precipitation for the day in band in HRU (mm H<sub>2</sub>O)*
- real \*8, dimension(:,:), allocatable [tavband](#)
  - average temperature for the day in band in HRU (deg C)*
- real \*8, dimension(:,:), allocatable [wat\\_phi1](#)
  - cross-sectional area of flow at bankfull depth ( $\text{m}^2$ )*
- real \*8, dimension(:,:), allocatable [wat\\_phi5](#)
  - flow rate when reach is at bankfull depth ( $\text{m}^3/\text{s}$ )*
- real \*8, dimension(:,:), allocatable [wat\\_phi6](#)
  - bottom width of main channel (m)*
- real \*8, dimension(:,:), allocatable [wat\\_phi9](#)
  - depth of water when reach is at bankfull depth (m)*
- real \*8, dimension(:,:), allocatable [snoeb](#)
  - snow water content in elevation band on current day (mm H<sub>2</sub>O)*
- real \*8, dimension(:,:), allocatable [wudeep](#)
  - average daily water removal from the deep aquifer for the month for the HRU within the subbasin ( $10^4 \text{ m}^3/\text{day}$ )*
- real \*8, dimension(:,:), allocatable [wushal](#)
  - average daily water removal from the shallow aquifer for the month for the HRU within the subbasin ( $10^4 \text{ m}^3/\text{day}$ )*
- real \*8, dimension(:,:), allocatable [bss](#)
  - bss(1,:) amount of lateral flow lagged (mm H<sub>2</sub>O)*
  - bss(2,:) amount of nitrate in lateral flow lagged (kg N/ha)*
  - bss(3,:) amount of tile flow lagged (mm)*
  - bss(4,:) amount of nitrate in tile flow lagged (kg N/ha)*
- real \*8, dimension(:,:), allocatable [nsetlw](#)

- nsetlw(1,:) nitrogen settling rate for 1st season (m/day)*
  - nsetlw(2,:) nitrogen settling rate for 2nd season (m/day)*
- real \*8, dimension(:,:), allocatable **snotmpeb**
  - temperature of snow pack in elevation band (deg C)*
- real \*8, dimension(:,:), allocatable **surf\_bs**
  - surf\_bs(1,:) amount of surface runoff lagged over one day (mm H2O)*
  - surf\_bs(2,:) amount of sediment yield lagged over one day (metric tons)*
  - surf\_bs(3,:) amount of organic nitrogen loading lagged over one day (kg N/ha)*
  - surf\_bs(4,:) amount of organic phosphorus loading lagged over one day (kg P/ha)*
  - surf\_bs(5,:) amount of nitrate loading in surface runoff lagged over one day (kg N/ha)*
  - surf\_bs(6,:) amount of soluble phosphorus loading lagged over one day (kg P/ha)*
  - surf\_bs(7,:) amount of active mineral phosphorus loading lagged over one day (kg P/ha)*
  - surf\_bs(8,:) amount of stable mineral phosphorus loading lagged over one day (kg P/ha)*
  - surf\_bs(9,:) amount of less persistent bacteria in solution lagged over one day (# colonies/ha)*
  - surf\_bs(10,:) amount of persistent bacteria in solution lagged over one day (# colonies/ha)*
  - surf\_bs(11,:) amount of less persistent bacteria sorbed lagged over one day (# colonies/ha)*
  - surf\_bs(12,:) amount of persistent bacteria sorbed lagged over one day (# colonies/ha)*
- real \*8, dimension(:,:), allocatable **nsetlp**
  - nsetlp(1,:) nitrogen settling rate for 1st season (m/day)*
  - nsetlp(2,:) nitrogen settling rate for 2nd season (m/day)*
- real \*8, dimension(:,:), allocatable **tmxband**
  - maximum temperature for the day in band in HRU (deg C)*
- real \*8, dimension(:,:), allocatable **fracd**
  - fraction of solar radiation occurring during hour in day in HRU (none)*
- real \*8, dimension(:,:), allocatable **rainsub**
  - precipitation for the time step during the day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable **rstopbsb**
- real \*8, dimension(:,:), allocatable **orig\_snoeb**
- real \*8, dimension(:,:), allocatable **orig\_pltpst**
- real \*8, dimension(:,:), allocatable **terr\_p**
- real \*8, dimension(:,:), allocatable **terr\_cn**
- real \*8, dimension(:,:), allocatable **terr\_sl**
- real \*8, dimension(:,:), allocatable **drain\_d**
- real \*8, dimension(:,:), allocatable **drain\_t**
- real \*8, dimension(:,:), allocatable **drain\_g**
- real \*8, dimension(:,:), allocatable **drain\_idep**
- real \*8, dimension(:,:), allocatable **cont\_cn**
- real \*8, dimension(:,:), allocatable **cont\_p**
- real \*8, dimension(:,:), allocatable **strip\_n**
- real \*8, dimension(:,:), allocatable **strip\_cn**
- real \*8, dimension(:,:), allocatable **strip\_p**
- real \*8, dimension(:,:), allocatable **fire\_cn**
- integer, dimension(:,:), allocatable **cropno\_upd**
- real \*8, dimension(:,:), allocatable **hi\_upd**
- real \*8, dimension(:,:), allocatable **laimx\_upd**
- real \*8, dimension(:,:), allocatable **pst\_lag**
  - pst\_lag(1,:,:) amount of soluble pesticide in surface runoff lagged (kg pst/ha)*
  - pst\_lag(2,:,:) amount of sorbed pesticide in surface runoff lagged (kg pst/ha)*
  - pst\_lag(3,:,:) amount of pesticide lagged (kg pst/ha)*
- integer, dimension(:), allocatable **hrupest**
  - pesticide use flag (none)*
  - 0: no pesticides used in HRU*
  - 1: pesticides used in HRU*
- integer, dimension(:), allocatable **swtrg**



- rainfall event flag (none):*
  - 0: no rainfall event over midnight*
  - 1: rainfall event over midnight*
- integer, dimension(:), allocatable **igro**
  - land cover status code (none). This code informs the model whether or not a land cover is growing at the beginning of the simulation*
  - 0 no land cover currently growing*
  - 1 land cover growing*
- integer, dimension(:,:), allocatable **ipnd**
  - ipnd(1,:) beginning month of 2nd "season" of nutrient settling season (none)*
  - ipnd(2,:) ending month of 2nd "season" of nutrient settling season (none)*
- integer, dimension(:,:), allocatable **iflod**
  - iflod(1,:) beginning month of non-flood season (none)*
  - iflod(2,:) ending month of non-flood season (none)*
- integer, dimension(:), allocatable **ndtarg**
  - number of days required to reach target storage from current pond storage (none)*
- integer, dimension(:), allocatable **nstress**
  - code for approach used to determine amount of nitrogen to HRU (none):*
  - 0 nitrogen target approach*
  - 1 annual max approach*
- integer, dimension(:), allocatable **iafrttyp**
- integer, dimension(:), allocatable **igrotree**
- integer, dimension(:), allocatable **grz\_days**
  - number of days grazing will be simulated (none)*
- integer, dimension(:), allocatable **nmgt**
  - management code (for GIS output only) (none)*
- integer, dimension(:), allocatable **icr**
  - sequence number of crop grown within the current year (none)*
- integer, dimension(:), allocatable **irrno**
  - irrigation source location (none)*
  - if IRRSC=1, IRRNO is the number of the reach*
  - if IRRSC=2, IRRNO is the number of the reservoir*
  - if IRRSC=3, IRRNO is the number of the subbasin*
  - if IRRSC=4, IRRNO is the number of the subbasin*
  - if IRRSC=5, not used*
- integer, dimension(:), allocatable **sol\_nly**
  - number of soil layers in HRU (none)*
- integer, dimension(:), allocatable **npcp**
  - prior day category (none)*
  - 1 dry day*
  - 2 wet day*
- integer, dimension(:), allocatable **irn**
  - average annual number of irrigation applications in HRU (none)*
- integer, dimension(:), allocatable **igrz**
  - grazing flag for HRU (none):*
  - 0 HRU currently not grazed*
  - 1 HRU currently grazed*
- integer, dimension(:), allocatable **ndeat**
  - number of days HRU has been grazed (days)*
- integer, dimension(:), allocatable **hru\_sub**
  - subbasin number in which HRU/reach is located (none)*
- integer, dimension(:), allocatable **urblu**
  - urban land type identification number from urban database (urban.dat) (none)*
- integer, dimension(:), allocatable **ldrain**



- soil layer where drainage tile is located (none)*
- integer, dimension(:), allocatable **idorm**
  - dormancy status code (none):*
  - 0 land cover growing (not dormant)*
  - 1 land cover dormant*
- integer, dimension(:), allocatable **hru\_seq**
- integer, dimension(:), allocatable **iurban**
  - urban simulation code (none):*
  - 0 no urban sections in HRU*
  - 1 urban sections in HRU, simulate using USGS regression equations*
  - 2 urban sections in HRU, simulate using build up/wash off algorithm*
- integer, dimension(:), allocatable **icftr**
  - continuous fertilizer flag for HRU (none):*
  - 0 HRU currently not continuously fertilized*
  - 1 HRU currently continuously fertilized*
- integer, dimension(:), allocatable **iday\_fert**
- integer, dimension(:), allocatable **hrgis**
  - GIS code printed to output files (output.hru, output.rch) (none)*
- integer, dimension(:), allocatable **ndcftr**
  - number of days HRU has been continuously fertilized (days)*
- integer, dimension(:), allocatable **irrsc**
  - irrigation source code (none):*
  - 1 divert water from reach*
  - 2 divert water from reservoir*
  - 3 divert water from shallow aquifer*
  - 4 divert water from deep aquifer*
  - 5 divert water from source outside watershed*
- integer, dimension(:), allocatable **orig\_igro**
- integer, dimension(:), allocatable **curyr\_mat**
- integer, dimension(:), allocatable **icpst**
  - icpst = 0 do not apply*
  - icpst = 1 application period*
- integer, dimension(:), allocatable **ndcpst**
  - current day within the application period (day)*
- integer, dimension(:), allocatable **iday\_pest**
  - current day between applications (day)*
- integer, dimension(:), allocatable **irr\_flag**
- integer, dimension(:,:), allocatable **rndseed**
  - random number generator seeds array. The seeds in the array are used to generate random numbers for the following purposes (none):*
  - (1) wet/dry day probability*
  - (2) solar radiation*
  - (3) precipitation*
  - (4) USLE rainfall erosion index*
  - (5) wind speed*
  - (6) 0.5 hr rainfall fraction*
  - (7) relative humidity*
  - (8) maximum temperature*
  - (9) minimum temperature*
  - (10) generate new random numbers*
- integer, dimension(:,:), allocatable **ncrops**
- integer, dimension(:), allocatable **manure\_id**
  - manure (fertilizer) identification number from fert.dat (none)*
- integer, dimension(:,:), allocatable **idplrot**
- integer, dimension(:,:), allocatable **iopday**
- integer, dimension(:,:), allocatable **iopyr**

- integer, dimension(:,:), allocatable **mgt\_ops**
- real \*8, dimension(:), allocatable **wshd\_pstap**  
*total or average annual amount of pesticide type applied in watershed during simulation (kg/ha)*
- real \*8, dimension(:), allocatable **wshd\_pstdg**  
*amount or average annual of pesticide lost through degradation in watershed (kg pst/ha)*
- integer, dimension(12) **ndmo**  
*cumulative number of days accrued in the month since the simulation began where the array location number is the number of the month (days)*
- integer, dimension(:), allocatable **npno**  
*array of unique pesticides used in watershed (none)*
- integer, dimension(:), allocatable **mcrhru**
- character(len=13), dimension(18) **rfile**  
*rainfall file names (.pcp)*
- character(len=13), dimension(18) **tfile**  
*temperature file names (.tmp)*
- character(len=4), dimension(1000) **urbname**  
*name of urban land use*
- character(len=1), dimension(:), allocatable **hyddgrp**
- character(len=16), dimension(:), allocatable **snam**  
*soil series name*
- character(len=17), dimension(300) **pname**  
*name of pesticide/toxin*
- character(len=4), dimension(60) **title**  
*title description lines in file.cio (1st 3 lines)*
- character(len=4), dimension(5000) **cpnm**  
*four character code to represent crop name*
- character(len=17), dimension(50) **fname**
- real \*8, dimension(:,:,:), allocatable **flomon**  
*average amount of water loaded to stream on a given day in the month ( $m^3/day$ )*
- real \*8, dimension(:,:,:), allocatable **solpstmon**  
*average daily soluble pesticide loading for month (mg pst/day)*
- real \*8, dimension(:,:,:), allocatable **srbspstmon**  
*average daily sorbed pesticide loading for month (mg pst/day)*
- real \*8, dimension(:,:,:), allocatable **orgnmon**  
*average amount of organic N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:,:,:), allocatable **orgpmon**  
*average amount of organic P loaded to stream on a given day in the month (kg P/day)*
- real \*8, dimension(:,:,:), allocatable **sedmon**  
*average amount of sediment loaded to stream on a given day in the month (metric tons/d)*
- real \*8, dimension(:,:,:), allocatable **minpmon**  
*average amount of soluble P loaded to stream on a given day in the month (kg P/day)*
- real \*8, dimension(:,:,:), allocatable **nh3mon**  
*average amount of NH3-N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:,:,:), allocatable **no3mon**  
*average amount of NO3-N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:,:,:), allocatable **bactlpmon**  
*average amount of less persistent bacteria loaded to stream on a given day in the month (# bact/day)*
- real \*8, dimension(:,:,:), allocatable **bactpmon**  
*average amount of persistent bacteria loaded to stream on a given day in the month (# bact/day)*
- real \*8, dimension(:,:,:), allocatable **no2mon**  
*average amount of NO2-N loaded to stream on a given day in the month (kg N/day)*

- real \*8, dimension(:,:,:), allocatable [cmtlmon](#)  
*cmtlmon(1,:,:) average amount of conservative metal #1 loaded to stream on a given day in the month (# bact/day)*  
*cmtlmon(2,:,:) average amount of conservative metal #2 loaded to stream on a given day in the month (# bact/day)*  
*cmtlmon(3,:,:) average amount of conservative metal #3 loaded to stream on a given day in the month (# bact/day)*
- real \*8, dimension(:,:), allocatable [cbodmon](#)  
*average amount of CBOD loaded to stream on a given day in the month (kg/day)*
- real \*8, dimension(:,:), allocatable [chlamon](#)  
*average amount of chlorophyll a loaded to stream on a given day in the month (kg/day)*
- real \*8, dimension(:,:), allocatable [disoxmon](#)  
*average amount of dissolved oxygen loaded to stream on a given day in the month (kg/day)*
- real \*8, dimension(:,:), allocatable [floyr](#)  
*average daily water loading for year ( $m^3/day$ )*
- real \*8, dimension(:,:), allocatable [orgnyr](#)  
*average daily organic N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [orgpyr](#)  
*average daily organic P loading for year (kg P/day)*
- real \*8, dimension(:,:), allocatable [sedyr](#)  
*average daily sediment loading for year (metric tons/day)*
- real \*8, dimension(:,:), allocatable [minpyr](#)  
*average daily mineral P loading for year (kg P/day)*
- real \*8, dimension(:,:), allocatable [nh3yr](#)  
*average daily NH3-N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [no2yr](#)  
*average daily NO2-N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [no3yr](#)  
*average daily NO3-N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [bactlpyr](#)  
*average daily loading of less persistent bacteria for year (# bact/day)*
- real \*8, dimension(:,:), allocatable [bactpyr](#)  
*average daily loading of persistent bacteria for year (# bact/day)*
- real \*8, dimension(:,:,:), allocatable [cmtlyr](#)  
*cmtlyr(1,:,:) average daily loading of conservative metal #1 for year (kg/day)*  
*cmtlyr(2,:,:) average daily loading of conservative metal #2 for year (kg/day)*  
*cmtlyr(3,:,:) average daily loading of conservative metal #3 for year (kg/day)*
- real \*8, dimension(:,:), allocatable [chlayr](#)  
*average daily loading of chlorophyll-a in year (kg/day)*
- real \*8, dimension(:,:), allocatable [cbodyr](#)  
*average daily loading of CBOD in year (kg/day)*
- real \*8, dimension(:,:), allocatable [disoxyr](#)  
*average daily loading of dissolved O2 in year (kg/day)*
- real \*8, dimension(:,:), allocatable [solpstyr](#)  
*average daily soluble pesticide loading for year (mg pst/day)*
- real \*8, dimension(:,:), allocatable [srbpstyr](#)  
*average daily sorbed pesticide loading for year (mg pst/day)*
- real \*8, dimension(:,:), allocatable [sol\\_mc](#)
- real \*8, dimension(:,:), allocatable [sol\\_mn](#)
- real \*8, dimension(:,:), allocatable [sol\\_mp](#)
- real \*8, dimension(:), allocatable [flocnst](#)  
*average daily water loading to reach ( $m^3 H_2O/day$ )*
- real \*8, dimension(:), allocatable [orgncnst](#)  
*average daily organic N loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [sedcnst](#)

- average daily sediment loading for reach (metric tons/day)*
- real \*8, dimension(:), allocatable [minpcnst](#)
  - average daily soluble P loading to reach (kg P/day)*
- real \*8, dimension(:), allocatable [no3cnst](#)
  - average daily nitrate loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [orgpcnst](#)
  - average daily organic P loading to reach (kg P/day)*
- real \*8, dimension(:), allocatable [bactpcnst](#)
  - average daily persistent bacteria loading to reach (# bact/day)*
- real \*8, dimension(:), allocatable [nh3cnst](#)
  - average daily ammonia loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [no2cnst](#)
  - average daily nitrite loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [bactlpcnst](#)
  - average daily less persistent bacteria loading to reach (# bact/day)*
- real \*8, dimension(:,:), allocatable [cmtlcnst](#)
  - cmtlcnst(1,:) average daily conservative metal #1 loading (kg/day)*
  - cmtlcnst(2,:) average daily conservative metal #2 loading (kg/day)*
  - cmtlcnst(3,:) average daily conservative metal #3 loading (kg/day)*
- real \*8, dimension(:), allocatable [chlacnst](#)
  - average daily chlorophyll-a loading to reach (kg/day)*
- real \*8, dimension(:), allocatable [disoxcnst](#)
  - average daily dissolved oxygen loading to reach (kg/day)*
- real \*8, dimension(:), allocatable [cbodcnst](#)
  - average daily loading of CBOD to reach (kg/day)*
- real \*8, dimension(:), allocatable [solpstcnst](#)
  - average daily soluble pesticide loading (mg/day)*
- real \*8, dimension(:), allocatable [srbspstcnst](#)
  - average daily sorbed pesticide loading (mg/day)*
- integer [nstep](#)
  - max number of time steps per day or number of lines of rainfall data for each day (depends on model operational time step) (none)*
- integer [idt](#)
  - length of time step used to report precipitation data for sub-daily modeling (operational time step) (minutes)*
- real \*8, dimension(:), allocatable [hdepth](#)
  - depth of flow during hour (m)*
- real \*8, dimension(:), allocatable [hhstor](#)
  - water stored in reach at end of hour ( $m^3$  H2O)*
- real \*8, dimension(:), allocatable [hrtwtr](#)
  - water leaving reach in hour ( $m^3$ )*
- real \*8, dimension(:), allocatable [hsdti](#)
  - flow rate in reach for hour ( $m^3/s$ )*
- real \*8, dimension(:), allocatable [hrchwtr](#)
  - water stored in reach at beginning of hour ( $m^3$  H2O)*
- real \*8, dimension(:), allocatable [nhh4](#)
  - ammonia concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [horgn](#)
  - organic nitrogen concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [halgae](#)
- real \*8, dimension(:), allocatable [hbod](#)
  - carbonaceous biochemical oxygen demand in reach at end of hour (mg O2/L)*

- real \*8, dimension(:), allocatable [hno2](#)  
*nitrite concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [hno3](#)  
*nitrate concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [horgp](#)  
*organic phosphorus concentration in reach at end of hour (mg P/L)*
- real \*8, dimension(:), allocatable [hsolp](#)  
*dissolved phosphorus concentration in reach at end of hour (mg P/L)*
- real \*8, dimension(:), allocatable [hchla](#)  
*chlorophyll-a concentration in reach at end of hour (mg chl-a/L)*
- real \*8, dimension(:), allocatable [hdisox](#)  
*dissolved oxygen concentration in reach at end of hour (mg O2/L)*
- real \*8, dimension(:), allocatable [hsedyld](#)  
*sediment transported out of reach during hour (metric tons)*
- real \*8, dimension(:), allocatable **hsedst**
- real \*8, dimension(:), allocatable [hharea](#)  
*cross-sectional area of flow ( $m^2$ )*
- real \*8, dimension(:), allocatable [hsolpst](#)  
*soluble pesticide concentration in outflow on day ( $mg\ pst/m^3$ )*
- real \*8, dimension(:), allocatable [hsorpst](#)  
*sorbed pesticide concentration in outflow on day ( $mg\ pst/m^3$ )*
- real \*8, dimension(:), allocatable [hhqday](#)  
*surface runoff generated each timestep of day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [precipdt](#)  
*precipitation, or effective precipitation reaching soil surface, in time step for HRU (mm H2O)*
- real \*8, dimension(:), allocatable [hhtime](#)  
*travel time of flow in reach for hour (hour)*
- real \*8, dimension(:), allocatable [hbactlp](#)  
*less persistent bacteria in reach/outflow during hour (# cfu/100mL)*
- real \*8, dimension(:), allocatable [hbactp](#)  
*persistent bacteria in reach/outflow during hour (# cfu/100mL)*
- integer, dimension(6) **ivar\_orig**
- real \*8, dimension(4) **rvar\_orig**
- integer **iatmodep**
- real \*8, dimension(:), allocatable **wattemp**
- real \*8, dimension(:), allocatable **lkpst\_mass**
- real \*8, dimension(:), allocatable **lkspst\_mass**
- real \*8, dimension(:), allocatable **vel\_chan**
- real \*8, dimension(:), allocatable [vfscn](#)  
*fraction of the total runoff from the entire field entering the most concentrated 10% of the VFS (none)*
- real \*8, dimension(:), allocatable [vfsratio](#)  
*field area/VFS area ratio (none)*
- real \*8, dimension(:), allocatable [vfsch](#)  
*fraction of flow entering the most concentrated 10% of the VFS which is fully channelized (none)*
- real \*8, dimension(:), allocatable **vfsi**
- real \*8, dimension(:,:), allocatable **filter\_i**
- real \*8, dimension(:,:), allocatable **filter\_ratio**
- real \*8, dimension(:,:), allocatable **filter\_con**
- real \*8, dimension(:,:), allocatable **filter\_ch**
- real \*8, dimension(:,:), allocatable **sol\_n**
- integer [cswat](#)

- = 0 Static soil carbon (old mineralization routines)*
- = 1 C-FARM one carbon pool model*
- = 2 Century model*
- real \*8, dimension(:,:), allocatable **tillagef**
- real \*8, dimension(:), allocatable **rtfr**
- real \*8, dimension(:), allocatable **stsol\_rd**  
*storing last soil root depth for use in harvestkilop/kilop (mm)*
- integer **dorm\_flag**
- real \*8 **bf\_flg**
- real \*8 **iabstr**
- real \*8, dimension(:), allocatable **ubntss**  
*TSS loading from urban impervious cover (metric tons)*
- real \*8, dimension(:), allocatable **ubnrunoff**  
*surface runoff from urban impervious cover (mm H2O)*
- real \*8, dimension(:,:), allocatable **sub\_ubnrunoff**  
*surface runoff from urban impervious cover in subbasin (mm H2O)*
- real \*8, dimension(:,:), allocatable **sub\_ubntss**  
*TSS loading from urban impervious cover in subbasin (metric tons)*
- real \*8, dimension(:,:,:), allocatable **hhsurf\_bs**
- integer **iuh**  
*unit hydrograph method: 1=triangular UH; 2=gamma function UH;*
- integer **sed\_ch**  
*channel routing for HOURLY; 0=Bagnold; 2=Brownlie; 3=Yang;*
- real \*8 **eros\_expo**  
*an exponent in the overland flow erosion equation ranges 1.5-3.0*
- real \*8 **eros\_spl**  
*coefficient of splash erosion varying 0.9-3.1*
- real \*8 **rill\_mult**  
*Multiplier to USLE\_K for soil susceptible to rill erosion, range 0.5-2.0.*
- real \*8 **c\_factor**
- real \*8 **ch\_d50**  
*median particle diameter of channel bed (mm)*
- real \*8 **sig\_g**  
*geometric standard deviation of particle sizes for the main channel. Mean air temperature at which precipitation is equally likely to be rain as snow/freezing rain.*
- real \*8 **uhalpha**  
*alpha coefficient for estimating unit hydrograph using a gamma function (\*.bsn)*
- real \*8 **abstinit**
- real \*8, dimension(:,:), allocatable **hhsedy**  
*sediment yield from HRU during a time step applied to HRU (tons)*
- real \*8, dimension(:,:), allocatable **sub\_subp\_dt**  
*precipitation for time step in subbasin (mm H2O)*
- real \*8, dimension(:,:), allocatable **sub\_hhsedy**  
*sediment yield for the time step in subbasin (metric tons)*
- real \*8, dimension(:,:), allocatable **sub\_atmp**
- real \*8, dimension(:), allocatable **rhy**  
*main channel hydraulic radius (m H2O)*
- real \*8, dimension(:), allocatable **hrtevp**  
*evaporation losses for hour ( $m^3$  H2O)*
- real \*8, dimension(:), allocatable **hrttlc**  
*transmission losses for hour ( $m^3$  H2O)*
- real \*8, dimension(:), allocatable **dratio**

- real \*8, dimension(:,:), allocatable **rchhr**
- real \*8, dimension(:), allocatable **hhresflwo**
- real \*8, dimension(:), allocatable **hhressedo**
- character(len=4), dimension(30) **lu\_nodrain**
- integer, dimension(:), allocatable **bmpdrain**
- real \*8, dimension(:), allocatable **sub\_cn2**
- real \*8, dimension(:), allocatable **sub\_ha\_urb**
- real \*8, dimension(:), allocatable **bmp\_recharge**
- real \*8, dimension(:), allocatable **sub\_ha\_imp**
- real \*8, dimension(:), allocatable **subdr\_km**
- real \*8, dimension(:), allocatable **subdr\_ickm**
- real \*8, dimension(:,:), allocatable **sf\_im**
- real \*8, dimension(:,:), allocatable **sf\_iy**
- real \*8, dimension(:,:), allocatable **sp\_sa**
- real \*8, dimension(:,:), allocatable **sp\_pvol**
- real \*8, dimension(:,:), allocatable **sp\_pd**
- real \*8, dimension(:,:), allocatable **sp\_sedi**
- real \*8, dimension(:,:), allocatable **sp\_sede**
- real \*8, dimension(:,:), allocatable **ft\_sa**
- real \*8, dimension(:,:), allocatable **ft\_fsa**
- real \*8, dimension(:,:), allocatable **ft\_dep**
- real \*8, dimension(:,:), allocatable **ft\_h**
- real \*8, dimension(:,:), allocatable **ft\_pd**
- real \*8, dimension(:,:), allocatable **ft\_k**
- real \*8, dimension(:,:), allocatable **ft\_dp**
- real \*8, dimension(:,:), allocatable **ft\_dc**
- real \*8, dimension(:,:), allocatable **ft\_por**
- real \*8, dimension(:,:), allocatable **tss\_den**
- real \*8, dimension(:,:), allocatable **ft\_alp**
- real \*8, dimension(:,:), allocatable **sf\_fr**
- real \*8, dimension(:,:), allocatable **sp\_qi**
- real \*8, dimension(:,:), allocatable **sp\_k**
- real \*8, dimension(:,:), allocatable **ft\_qpnd**
- real \*8, dimension(:,:), allocatable **sp\_dp**
- real \*8, dimension(:,:), allocatable **ft\_qsw**
- real \*8, dimension(:,:), allocatable **ft\_qin**
- real \*8, dimension(:,:), allocatable **ft\_qout**
- real \*8, dimension(:,:), allocatable **ft\_sedpnd**
- real \*8, dimension(:,:), allocatable **sp\_bpw**
- real \*8, dimension(:,:), allocatable **ft\_bpw**
- integer, dimension(:), allocatable **num\_sf**
- integer, dimension(:,:), allocatable **sf\_typ**
- integer, dimension(:,:), allocatable **sf\_dim**
- integer, dimension(:,:), allocatable **ft\_qfg**
- integer, dimension(:,:), allocatable **sp\_qfg**
- integer, dimension(:,:), allocatable **sf\_ptp**
- real \*8, dimension(:,:), allocatable **ft\_fc**
- real \*8 **sfsedmean**
- real \*8 **sfsedstdev**
- integer, dimension(:), allocatable **dtp\_imo**  
*month the reservoir becomes operational (none)*
- integer, dimension(:), allocatable **dtp\_iyr**  
*year of the simulation that the reservoir becomes operational (none)*
- integer, dimension(:), allocatable **dtp\_numstage**

- total number of stages in the weir (none)*
- integer, dimension(:), allocatable [dtp\\_numweir](#)
  - total number of weirs in the BMP (none)*
- integer, dimension(:), allocatable [dtp\\_onoff](#)
  - sub-basin detention pond is associated with (none)*
- integer, dimension(:), allocatable [dtp\\_reltype](#)
  - equations for stage-discharge relationship (none):*
  - 1=exponential function,*
  - 2=linear,*
  - 3=logarithmic,*
  - 4=cubic,*
  - 5=power*
- integer, dimension(:), allocatable [dtp\\_stagdis](#)
  - (none):*
  - 0=use weir/orifice discharge equation to calculate outflow,*
  - 1=use stage-dicharge relationship*
- real \*8, dimension(:), allocatable [cf](#)
  - this parameter controls the response of decomposition to the combined effect of soil temperature and moisture.*
- real \*8, dimension(:), allocatable [cfh](#)
  - maximum humification rate*
- real \*8, dimension(:), allocatable [cfdec](#)
  - the undisturbed soil turnover rate under optimum soil water and temperature. Increasing it will increase carbon and organic N decomp.*
- real \*8, dimension(:), allocatable [lat\\_orgn](#)
- real \*8, dimension(:), allocatable [lat\\_orgp](#)
- integer, dimension(:,:), allocatable [dtp\\_weirdim](#)
  - weir dimensions (none),*
  - 1=read user input,*
  - 0=use model calculation*
- integer, dimension(:,:), allocatable [dtp\\_weirtype](#)
  - type of weir (none):*
  - 1=rectangular and*
  - 2=circular*
- real \*8, dimension(:,:), allocatable [dtp\\_coef](#)
  - dtp\_coef(1,:) coefficient of 3rd degree in the polynomial equation (none)*
  - dtp\_coef(2,:) coefficient of 2nd degree in the polynomial equation (none)*
  - dtp\_coef(3,:) coefficient of 1st degree in the polynomial equation (none)*
- real \*8, dimension(:), allocatable [dtp\\_evrsv](#)
  - detention pond evaporation coefficient (none)*
- real \*8, dimension(:), allocatable [dtp\\_expont](#)
  - exponent used in the exponential equation (none)*
- real \*8, dimension(:), allocatable [dtp\\_intcept](#)
  - intercept used in regression equations (none)*
- real \*8, dimension(:), allocatable [dtp\\_lwratio](#)
  - ratio of length to width of water back up (none)*
- real \*8, dimension(:), allocatable [dtp\\_totwrwid](#)
  - total constructed width of the detention wall across the creek (m)*
- real \*8, dimension(:), allocatable [dtp\\_ivol](#)
- real \*8, dimension(:), allocatable [dtp\\_ised](#)
- integer, dimension(:,:), allocatable [ro\\_bmp\\_flag](#)
- real \*8, dimension(:,:), allocatable [psp\\_store](#)
- real \*8, dimension(:,:), allocatable [ssp\\_store](#)
- real \*8, dimension(:,:), allocatable [sol\\_cal](#)
- real \*8, dimension(:,:), allocatable [sol\\_ph](#)



- integer **sol\_p\_model**
- integer, dimension(:,:), allocatable **a\_days**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_flo**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_sed**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_bac**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_pp**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_sp**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_pn**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_sn**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_flos**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_seds**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_bacs**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_pps**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_sps**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_pns**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_sns**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_flot**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_sedt**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_bact**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_ppt**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_spt**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_pnt**
- real \*8, dimension(:,:), allocatable **ro\_bmp\_snt**
- real \*8, dimension(:), allocatable **bmp\_flo**
- real \*8, dimension(:), allocatable **bmp\_sed**
- real \*8, dimension(:), allocatable **bmp\_bac**
- real \*8, dimension(:), allocatable **bmp\_pp**
- real \*8, dimension(:), allocatable **bmp\_sp**
- real \*8, dimension(:), allocatable **bmp\_pn**
- real \*8, dimension(:), allocatable **bmp\_sn**
- real \*8, dimension(:), allocatable **bmp\_flag**
- real \*8, dimension(:), allocatable **bmp\_flos**
- real \*8, dimension(:), allocatable **bmp\_seds**
- real \*8, dimension(:), allocatable **bmp\_bacs**
- real \*8, dimension(:), allocatable **bmp\_pps**
- real \*8, dimension(:), allocatable **bmp\_sps**
- real \*8, dimension(:), allocatable **bmp\_pns**
- real \*8, dimension(:), allocatable **bmp\_sns**
- real \*8, dimension(:), allocatable **bmp\_flot**
- real \*8, dimension(:), allocatable **bmp\_sedt**
- real \*8, dimension(:), allocatable **bmp\_bact**
- real \*8, dimension(:), allocatable **bmp\_ppt**
- real \*8, dimension(:), allocatable **bmp\_spt**
- real \*8, dimension(:), allocatable **bmp\_pnt**
- real \*8, dimension(:), allocatable **bmp\_snt**
- real \*8, dimension(:,:), allocatable **ntp\_addon**  
*the distance between spillway levels (m)*
- real \*8, dimension(:,:), allocatable **ntp\_cdis**  
*discharge coefficient for weir/orifice flow at different stages (none)*
- real \*8, dimension(:,:), allocatable **ntp\_depweir**  
*depth of rectangular weir at different stages (m)*
- real \*8, dimension(:,:), allocatable **ntp\_dia weir**  
*diameter of circular weir at different stages (m)*
- real \*8, dimension(:,:), allocatable **ntp\_flowrate**

- maximum discharge from each stage of the weir/hole ( $m^3/s$ )*
  - real \*8, dimension(:,:), allocatable [dtp\\_pcpnet](#)
  - precipitation for different return periods (not used) (mm)*
  - real \*8, dimension(:,:), allocatable [dtp\\_retperd](#)
  - return period at different stages (years)*
  - real \*8, dimension(:,:), allocatable [dtp\\_wdratio](#)
  - width depth ratio of rectangular weirs at different stages (none)*
  - real \*8, dimension(:,:), allocatable **dtp\_wrwid**
  - real \*8, dimension(:), allocatable **ri\_subkm**
  - real \*8, dimension(:), allocatable **ri\_totpv**
  - real \*8, dimension(:), allocatable **irmmdt**
  - real \*8, dimension(:,:), allocatable [ri\\_sed](#)
  - total sediment deposited in the pond (tons)*
  - real \*8, dimension(:,:), allocatable **ri\_fr**
  - real \*8, dimension(:,:), allocatable **ri\_dim**
  - real \*8, dimension(:,:), allocatable **ri\_im**
  - real \*8, dimension(:,:), allocatable **ri\_iy**
  - real \*8, dimension(:,:), allocatable **ri\_sa**
  - real \*8, dimension(:,:), allocatable **ri\_vol**
  - real \*8, dimension(:,:), allocatable **ri\_qi**
  - real \*8, dimension(:,:), allocatable **ri\_k**
  - real \*8, dimension(:,:), allocatable **ri\_dd**
  - real \*8, dimension(:,:), allocatable **ri\_evsv**
  - real \*8, dimension(:,:), allocatable **ri\_dep**
  - real \*8, dimension(:,:), allocatable **ri\_ndt**
  - real \*8, dimension(:,:), allocatable **ri\_pmpvol**
  - real \*8, dimension(:,:), allocatable **ri\_sed\_cumul**
  - real \*8, dimension(:,:), allocatable **hrnopcp**
  - real \*8, dimension(:,:), allocatable **ri\_qloss**
  - real \*8, dimension(:,:), allocatable **ri\_pumpv**
  - real \*8, dimension(:,:), allocatable **ri\_sedi**
  - character(len=4), dimension(:,:), allocatable **ri\_nirr**
  - integer, dimension(:), allocatable **num\_ri**
  - integer, dimension(:), allocatable **ri\_luflg**
  - integer, dimension(:), allocatable **num\_noirr**
  - integer, dimension(:), allocatable **wtp\_subnum**
  - integer, dimension(:), allocatable **wtp\_onoff**
  - integer, dimension(:), allocatable **wtp\_imo**
  - integer, dimension(:), allocatable **wtp\_iyr**
  - integer, dimension(:), allocatable **wtp\_dim**
  - integer, dimension(:), allocatable **wtp\_stagdis**
  - integer, dimension(:), allocatable **wtp\_sdtype**
  - real \*8, dimension(:), allocatable [wtp\\_evsv](#)
  - detention pond evaporation coefficient (none)*
  - real \*8, dimension(:), allocatable [wtp\\_pvol](#)
  - volume of permanent pool including forebay ( $m^3 H_2O$ )*
  - real \*8, dimension(:), allocatable **wtp\_pdepth**
  - real \*8, dimension(:), allocatable **wtp\_sdslope**
  - real \*8, dimension(:), allocatable **wtp\_lenwidth**
  - real \*8, dimension(:), allocatable **wtp\_extdepth**
  - real \*8, dimension(:), allocatable **wtp\_hydeff**
  - real \*8, dimension(:), allocatable **wtp\_sdintc**
  - real \*8, dimension(:), allocatable **wtp\_sdexp**
  - real \*8, dimension(:), allocatable **wtp\_sdc1**

- real \*8, dimension(:), allocatable **wtp\_sdc2**
- real \*8, dimension(:), allocatable **wtp\_sdc3**
- real \*8, dimension(:), allocatable **wtp\_pdia**
- real \*8, dimension(:), allocatable **wtp\_plen**
- real \*8, dimension(:), allocatable **wtp\_pmann**
- real \*8, dimension(:), allocatable **wtp\_ploss**
- real \*8, dimension(:), allocatable **wtp\_k**
- real \*8, dimension(:), allocatable **wtp\_dp**
- real \*8, dimension(:), allocatable **wtp\_sedi**
- real \*8, dimension(:), allocatable **wtp\_sede**
- real \*8, dimension(:), allocatable **wtp\_qi**
- real \*8 **lai\_init**  
*initial leaf area index of transplants*
- real \*8 **bio\_init**  
*initial biomass of transplants (kg/ha)*
- real \*8 **cnop**  
*SCS runoff curve number for moisture condition II (none)*
- real \*8 **harveff**  
*harvest efficiency: fraction of harvested yield that is removed from HRU; the remainder becomes residue on the soil surface(none)*
- real \*8 **hi\_ovr**  
*harvest index target specified at harvest ((kg/ha)/(kg/ha))*
- real \*8 **frac\_harvk**
- real \*8 **lid\_vgcl**  
*van Genuchten equation's coefficient, l (none)*
- real \*8 **lid\_vgcm**  
*van Genuchten equation's coefficient, m (none)*
- real \*8 **lid\_qsurf\_total**
- real \*8 **lid\_farea\_sum**
- real \*8, dimension(:,:), allocatable **lid\_cuminf\_last**  
*cumulative amount of water infiltrated into the amended soil layer at the last time step in a day (mm H2O)*
- real \*8, dimension(:,:), allocatable **lid\_cumr\_last**  
*cumulative amount of rainfall at the last time step in a day (mm H2O)*
- real \*8, dimension(:,:), allocatable **lid\_excum\_last**  
*cumulative amount of excess rainfall at the last time step in a day (mm H2O)*
- real \*8, dimension(:,:), allocatable **lid\_f\_last**  
*potential infiltration rate of the amended soil layer at the last time step in a day (mm/mm H2O)*
- real \*8, dimension(:,:), allocatable **lid\_sw\_last**  
*soil water content of the amended soil layer at the last time step in a day (mm/mm H2O)*
- real \*8, dimension(:,:), allocatable **lid\_qsurf**  
*depth of runoff generated on a LID in a given time interval (mm H2O)*
- real \*8, dimension(:,:), allocatable **interval\_last**
- real \*8, dimension(:,:), allocatable **lid\_str\_last**
- real \*8, dimension(:,:), allocatable **lid\_farea**
- real \*8, dimension(:,:), allocatable **lid\_sw\_add**
- real \*8, dimension(:,:), allocatable **lid\_cumqperc\_last**
- real \*8, dimension(:,:), allocatable **lid\_cumirr\_last**
- integer, dimension(:,:), allocatable **gr\_onoff**
- integer, dimension(:,:), allocatable **gr\_imo**
- integer, dimension(:,:), allocatable **gr\_iyr**
- real \*8, dimension(:,:), allocatable **gr\_farea**  
*fractional area of a green roof to the HRU (none)*

- real \*8, dimension(:,:), allocatable **gr\_solop**
- real \*8, dimension(:,:), allocatable **gr\_etcoef**
- real \*8, dimension(:,:), allocatable **gr\_fc**
- real \*8, dimension(:,:), allocatable **gr\_wp**
- real \*8, dimension(:,:), allocatable **gr\_ksat**
- real \*8, dimension(:,:), allocatable **gr\_por**
- real \*8, dimension(:,:), allocatable **gr\_hydeff**
- real \*8, dimension(:,:), allocatable **gr\_soldpt**
- integer, dimension(:,:), allocatable **rg\_onoff**
- integer, dimension(:,:), allocatable **rg\_imo**
- integer, dimension(:,:), allocatable **rg\_iyr**
- real \*8, dimension(:,:), allocatable **rg\_farea**
- real \*8, dimension(:,:), allocatable **rg\_solop**
- real \*8, dimension(:,:), allocatable **rg\_etcoef**
- real \*8, dimension(:,:), allocatable **rg\_fc**
- real \*8, dimension(:,:), allocatable **rg\_wp**
- real \*8, dimension(:,:), allocatable **rg\_ksat**
- real \*8, dimension(:,:), allocatable **rg\_por**
- real \*8, dimension(:,:), allocatable **rg\_hydeff**
- real \*8, dimension(:,:), allocatable **rg\_soldpt**
- real \*8, dimension(:,:), allocatable **rg\_dimop**
- real \*8, dimension(:,:), allocatable **rg\_sarea**
- real \*8, dimension(:,:), allocatable **rg\_vol**
- real \*8, dimension(:,:), allocatable **rg\_sth**
- real \*8, dimension(:,:), allocatable **rg\_sdia**
- real \*8, dimension(:,:), allocatable **rg\_bdia**
- real \*8, dimension(:,:), allocatable **rg\_sts**
- real \*8, dimension(:,:), allocatable **rg\_orifice**
- real \*8, dimension(:,:), allocatable **rg\_oheight**
- real \*8, dimension(:,:), allocatable **rg\_odia**
- integer, dimension(:,:), allocatable **cs\_onoff**
- integer, dimension(:,:), allocatable **cs\_imo**
- integer, dimension(:,:), allocatable **cs\_iyr**
- integer, dimension(:,:), allocatable **cs\_grcon**
- real \*8, dimension(:,:), allocatable **cs\_farea**
- real \*8, dimension(:,:), allocatable **cs\_vol**
- real \*8, dimension(:,:), allocatable **cs\_rdepth**
- integer, dimension(:,:), allocatable **pv\_onoff**
- integer, dimension(:,:), allocatable **pv\_imo**
- integer, dimension(:,:), allocatable **pv\_iyr**
- integer, dimension(:,:), allocatable **pv\_solop**
- real \*8, dimension(:,:), allocatable **pv\_grvdep**
- real \*8, dimension(:,:), allocatable **pv\_grvpor**
- real \*8, dimension(:,:), allocatable **pv\_farea**
- real \*8, dimension(:,:), allocatable **pv\_drcoef**
- real \*8, dimension(:,:), allocatable **pv\_fc**
- real \*8, dimension(:,:), allocatable **pv\_wp**
- real \*8, dimension(:,:), allocatable **pv\_ksat**
- real \*8, dimension(:,:), allocatable **pv\_por**
- real \*8, dimension(:,:), allocatable **pv\_hydeff**
- real \*8, dimension(:,:), allocatable **pv\_soldpt**
- integer, dimension(:,:), allocatable **lid\_onoff**
- real \*8, dimension(:,:), allocatable **sol\_hsc**

*mass of C present in slow humus (kg ha<sup>-1</sup>)*

- real \*8, dimension(:,:), allocatable **sol\_hsn**  
*mass of N present in slow humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_hpc**  
*mass of C present in passive humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_hpn**  
*mass of N present in passive humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_lm**  
*mass of metabolic litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_lmc**  
*mass of C in metabolic litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_lmn**  
*mass of N in metabolic litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_ls**  
*mass of structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_lsc**  
*mass of C in structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_lsl**  
*mass of lignin in structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_lsn**  
*mass of N in structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **sol\_bmc**
- real \*8, dimension(:,:), allocatable **sol\_bmn**
- real \*8, dimension(:,:), allocatable **sol\_rnmn**
- real \*8, dimension(:,:), allocatable **sol\_lslc**
- real \*8, dimension(:,:), allocatable **sol\_lslnc**
- real \*8, dimension(:,:), allocatable **sol\_rspc**
- real \*8, dimension(:,:), allocatable **sol\_woc**
- real \*8, dimension(:,:), allocatable **sol\_won**
- real \*8, dimension(:,:), allocatable **sol\_hp**
- real \*8, dimension(:,:), allocatable **sol\_hs**
- real \*8, dimension(:,:), allocatable **sol\_bm**
- real \*8, dimension(:,:), allocatable **sol\_cac**
- real \*8, dimension(:,:), allocatable **sol\_cec**
- real \*8, dimension(:,:), allocatable **sol\_percc**
- real \*8, dimension(:,:), allocatable **sol\_latc**
- real \*8, dimension(:), allocatable **sedc\_d**  
*amount of C lost with sediment pools (kg C/ha)*
- real \*8, dimension(:), allocatable **surfqc\_d**
- real \*8, dimension(:), allocatable **latc\_d**
- real \*8, dimension(:), allocatable **percc\_d**
- real \*8, dimension(:), allocatable **foc\_d**
- real \*8, dimension(:), allocatable **nppc\_d**
- real \*8, dimension(:), allocatable **rsdc\_d**
- real \*8, dimension(:), allocatable **grainc\_d**
- real \*8, dimension(:), allocatable **stoverc\_d**
- real \*8, dimension(:), allocatable **soc\_d**
- real \*8, dimension(:), allocatable **rspc\_d**
- real \*8, dimension(:), allocatable **emitc\_d**
- real \*8, dimension(:), allocatable **sub\_sedc\_d**
- real \*8, dimension(:), allocatable **sub\_surfqc\_d**
- real \*8, dimension(:), allocatable **sub\_latc\_d**
- real \*8, dimension(:), allocatable **sub\_percc\_d**
- real \*8, dimension(:), allocatable **sub\_foc\_d**

- real \*8, dimension(:), allocatable **sub\_nppc\_d**
- real \*8, dimension(:), allocatable **sub\_rsd\_c\_d**
- real \*8, dimension(:), allocatable **sub\_grainc\_d**
- real \*8, dimension(:), allocatable **sub\_stoverc\_d**
- real \*8, dimension(:), allocatable **sub\_emitc\_d**
- real \*8, dimension(:), allocatable **sub\_soc\_d**
- real \*8, dimension(:), allocatable **sub\_rspc\_d**
- real \*8, dimension(:), allocatable **sedc\_m**
- real \*8, dimension(:), allocatable **surfqc\_m**
- real \*8, dimension(:), allocatable **latc\_m**
- real \*8, dimension(:), allocatable **percc\_m**
- real \*8, dimension(:), allocatable **foc\_m**
- real \*8, dimension(:), allocatable **nppc\_m**
- real \*8, dimension(:), allocatable **rsdc\_m**
- real \*8, dimension(:), allocatable **grainc\_m**
- real \*8, dimension(:), allocatable **stoverc\_m**
- real \*8, dimension(:), allocatable **emitc\_m**
- real \*8, dimension(:), allocatable **soc\_m**
- real \*8, dimension(:), allocatable **rspc\_m**
- real \*8, dimension(:), allocatable **sedc\_a**
- real \*8, dimension(:), allocatable **surfqc\_a**
- real \*8, dimension(:), allocatable **latc\_a**
- real \*8, dimension(:), allocatable **percc\_a**
- real \*8, dimension(:), allocatable **foc\_a**
- real \*8, dimension(:), allocatable **nppc\_a**
- real \*8, dimension(:), allocatable **rsdc\_a**
- real \*8, dimension(:), allocatable **grainc\_a**
- real \*8, dimension(:), allocatable **stoverc\_a**
- real \*8, dimension(:), allocatable **emitc\_a**
- real \*8, dimension(:), allocatable **soc\_a**
- real \*8, dimension(:), allocatable **rspc\_a**
- integer, dimension(:), allocatable **tillage\_switch**
- real \*8, dimension(:), allocatable **tillage\_depth**
- integer, dimension(:), allocatable **tillage\_days**
- real \*8, dimension(:), allocatable **tillage\_factor**
- real \*8 **dthy**

*time interval for subdaily flood routing*

- integer, dimension(4) **ihx**
- integer, dimension(:), allocatable **nhy**
- real \*8, dimension(:), allocatable **rchx**
- real \*8, dimension(:), allocatable **rcss**
- real \*8, dimension(:), allocatable **qcap**
- real \*8, dimension(:), allocatable **chxa**
- real \*8, dimension(:), allocatable **chxp**
- real \*8, dimension(:, :, :), allocatable **qhy**
- real \*8 **ff1**
- real \*8 **ff2**

#### 4.1.1 Detailed Description

main module containing the global variables

## 4.1.2 Variable Documentation

### 4.1.2.1 igropt

integer parm::igropt

Qual2E option for calculating the local specific growth rate of algae

1: multiplicative.

$$u = mumax fll f_{nn} f_{pp}$$

2: limiting nutrient

$$u = mumax fll \min(f_{nn}, f_{pp})$$

3: harmonic mean

$$u = mumax fll \frac{2}{\frac{1}{f_{nn}} + \frac{1}{f_{pp}}}$$





# Chapter 5

## File Documentation

### 5.1 addh.f90 File Reference

#### Functions/Subroutines

- subroutine `addh` (*j*)  
*this subroutine adds loadings from two sources for routing*

#### 5.1.1 Detailed Description

file containing the subroutine `addh`

Author

modified by Javier Burguete

#### 5.1.2 Function/Subroutine Documentation

##### 5.1.2.1 `addh()`

```
subroutine addh (  
    integer, intent(in) j )
```

this subroutine adds loadings from two sources for routing

Parameters

<i>in</i>	<i>j</i>	hydrograph storage location number of first dataset to be added (none)
-----------	----------	--

## 5.2 albedo.f90 File Reference

### Functions/Subroutines

- subroutine [albedo](#) (j)  
*this subroutine calculates albedo in the HRU for the day*

#### 5.2.1 Detailed Description

file containing the subroutine albedo

##### Author

modified by Javier Burguete

#### 5.2.2 Function/Subroutine Documentation

##### 5.2.2.1 albedo()

```
subroutine albedo (
    integer, intent(in) j )
```

this subroutine calculates albedo in the HRU for the day

##### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.3 allocate\_parms.f90 File Reference

### Functions/Subroutines

- subroutine [allocate\\_parms](#)  
*this subroutine allocates array sizes*

#### 5.3.1 Detailed Description

file containing the subroutine allocate\_parms

##### Author

modified by Javier Burguete

## 5.4 alph.f90 File Reference

### Functions/Subroutines

- subroutine [alph](#) (iwave, j)

*this subroutine computes alpha, a dimensionless parameter that expresses the fraction of total rainfall that occurs during 0.5h*

#### 5.4.1 Detailed Description

file containing the subroutine alph

Author

modified by Javier Burguete

#### 5.4.2 Function/Subroutine Documentation

##### 5.4.2.1 alph()

```
subroutine alph (
    integer, intent(in) iwave,
    integer, intent(in) j )
```

this subroutine computes alpha, a dimensionless parameter that expresses the fraction of total rainfall that occurs during 0.5h

##### Parameters

in	iwave	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU MUSLE(sedyld) each hru is calculated independently using hru area and adjusted channel length iwave = 1 subbasin # for subbasin MUSLE is computed for entire subbasin using hru weighted KLSCP
in	j	HRU number

## 5.5 anfert.f90 File Reference

### Functions/Subroutines

- subroutine [anfert](#) (j)

*this subroutine automatically applies Nitrogen and Phosphorus when Nitrogen stress exceeds a user input threshold*

### 5.5.1 Detailed Description

file containing the subroutine anfert

Author

modified by Javier Burguete

### 5.5.2 Function/Subroutine Documentation

#### 5.5.2.1 anfert()

```
subroutine anfert (
    integer, intent(in) j )
```

this subroutine automatically applies Nitrogen and Phosphorus when Nitrogen stress exceeds a user input threshold

Parameters

in	j	HRU number
----	---	------------

## 5.6 apex\_day.f90 File Reference

### Functions/Subroutines

- subroutine [apex\\_day](#) (i, k)  
*this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a daily basis*

#### 5.6.1 Detailed Description

file containing the subroutine apex\_day

Author

modified by Javier Burguete

#### 5.6.2 Function/Subroutine Documentation

### 5.6.2.1 apex\_day()

```
subroutine apex_day (
    integer, intent(in) i,
    integer, intent(in) k )
```

this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a daily basis

#### Parameters

in	<i>i</i>	current day in simulation—loop counter (julian date)
in	<i>k</i>	reach number or file number (none)

## 5.7 apply.f90 File Reference

### Functions/Subroutines

- subroutine [apply](#) (*j*)  
*this subroutine applies pesticide*

### 5.7.1 Detailed Description

file containing the subroutine apply

#### Author

modified by Javier Burguete

### 5.7.2 Function/Subroutine Documentation

#### 5.7.2.1 apply()

```
subroutine apply (
    integer, intent(in) j )
```

this subroutine applies pesticide

#### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.8 ascrv.f90 File Reference

### Functions/Subroutines

- subroutine [ascrv](#) (x1, x2, x3, x4, x5, x6)

*this subroutine computes shape parameters x5 and x6 for the S curve equation*

#### 5.8.1 Detailed Description

file containing the subroutine ascrv

Author

modified by Javier Burguete

#### 5.8.2 Function/Subroutine Documentation

##### 5.8.2.1 ascrv()

```
subroutine ascrv (
    real*8, intent(in) x1,
    real*8, intent(in) x2,
    real*8, intent(in) x3,
    real*8, intent(in) x4,
    real*8, intent(out) x5,
    real*8, intent(out) x6 )
```

this subroutine computes shape parameters x5 and x6 for the S curve equation

$$x = \frac{y}{y + \exp(x5 + x6 y)}$$

given 2 (x,y) points along the curve. x5 is determined by solving the equation with  $x$  and  $y$  values measured around the midpoint of the curve (approx. 50% of the maximum value for  $x$ ) and x6 is determined by solving the equation with  $x$  and  $y$  values measured close to one of the endpoints of the curve (100% of the maximum value for  $x$ ). This subroutine is called from [readbsn.f90](#) and [readplant.f90](#)

##### Parameters

in	x1	value for x in the above equation for first datapoint, x1 should be close to 0.5 (the midpoint of the curve)
in	x2	value for x in the above equation for second datapoint, x2 should be close to 0.0 or 1.0
in	x3	value for y in the above equation corresponding to x1
in	x4	value for y in the above equation corresponding to x2
out	x5	1st shape parameter for S curve equation characterizing the midpoint of the curve
out	x6	2nd shape parameter for S curve equation characterizing the regions close to the endpoints of the curve

## 5.9 atri.f90 File Reference

### Functions/Subroutines

- real \*8 function [atri](#) (at1, at2, at3, at4i)

*this function generates a random number from a triangular distribution given X axis points at start, end, and peak Y value*

#### 5.9.1 Detailed Description

file containing the function atri

Author

modified by Javier Burguete

#### 5.9.2 Function/Subroutine Documentation

##### 5.9.2.1 atri()

```
real*8 function atri (
    real*8, intent(in) at1,
    real*8, intent(in) at2,
    real*8, intent(in) at3,
    integer, intent(inout) at4i )
```

this function generates a random number from a triangular distribution given X axis points at start, end, and peak Y value

##### Parameters

in	<i>at1</i>	lower limit for distribution (none)
in	<i>at2</i>	monthly mean for distribution (none)
in	<i>at3</i>	upper limit for distribution (none)
in, out	<i>at4i</i>	random number seed (none)

##### Returns

daily value generated for distribution (none)

## 5.10 aunif.f90 File Reference

### Functions/Subroutines

- real \*8 function [aunif](#) (x1)

*This function generates random numbers ranging from 0.0 to 1.0. In the process of calculating the random number, the seed (x1) is set to a new value. This function implements the prime-modulus generator.*

### 5.10.1 Detailed Description

file containing the function aunif

Author

modified by Javier Burguete

### 5.10.2 Function/Subroutine Documentation

#### 5.10.2.1 aunif()

```
real*8 function aunif (
    integer, intent(inout) x1 )
```

This function generates random numbers ranging from 0.0 to 1.0. In the process of calculating the random number, the seed (x1) is set to a new value. This function implements the prime-modulus generator.

$$xi = 16807 xi \bmod (2^{31} - 1)$$

using code which ensures that no intermediate result uses more than 31 bits. The theory behind the code is summarized in [1]

Parameters

in, out	x1	random number generator seed (integer) where $0 < x1 < 2147483647$
---------	----	--

Returns

random number ranging from 0.0 to 1.0

## 5.11 autoirr.f90 File Reference

### Functions/Subroutines

- subroutine [autoirr](#) (j)  
*this subroutine performs the auto-irrigation operation*

#### 5.11.1 Detailed Description

file containing the subroutine autoirr

Author

modified by Javier Burguete



## 5.11.2 Function/Subroutine Documentation

### 5.11.2.1 autoirr()

```
subroutine autoirr (
    integer, intent(in) j )
```

this subroutine performs the auto-irrigation operation

#### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.12 bacteria.f90 File Reference

### Functions/Subroutines

- subroutine [bacteria](#) (*j*)  
*this subroutine calculates bacteria growth, transport with runoff and loss due to percolation into soil*

### 5.12.1 Detailed Description

file containing the subroutine bacteria

#### Author

modified by Javier Burguete

## 5.12.2 Function/Subroutine Documentation

### 5.12.2.1 bacteria()

```
subroutine bacteria (
    integer, intent(in) j )
```

this subroutine calculates bacteria growth, transport with runoff and loss due to percolation into soil

#### Parameters

in	<i>j</i>	HRU number (none)
----	----------	-------------------

## 5.13 biozone.f90 File Reference

### Functions/Subroutines

- subroutine [biozone](#) (j)

*this subroutine conducts biophysical processes occuring in the biozone layer of a septic HRU. Septic algorithm adapted from [4]*

#### 5.13.1 Detailed Description

file containing the subroutine biozone

##### Author

J. Jeong,  
C. Santhi,  
modified by Javier Burguete

#### 5.13.2 Function/Subroutine Documentation

##### 5.13.2.1 biozone()

```
subroutine biozone (
    integer, intent(in) j )
```

this subroutine conducts biophysical processes occuring in the biozone layer of a septic HRU. Septic algorithm adapted from [4]

##### Parameters

<i>j</i>	HRU number
----------	------------

## 5.14 bmp\_det\_pond.f90 File Reference

### Functions/Subroutines

- subroutine [bmp\\_det\\_pond](#) (sb)

*the purpose of this subroutine is to read in data from the detention pond input file (.dtp) and perform computations*

#### 5.14.1 Detailed Description

file containing the subroutine bmp\_det\_pond

## Author

modified by Javier Burguete

## 5.14.2 Function/Subroutine Documentation

### 5.14.2.1 bmp\_det\_pond()

```
subroutine bmp_det_pond (
    integer, intent(in) sb )
```

the purpose of this subroutine is to read in data from the detention pond input file (.dtp) and perform computations

## Parameters

in	sb	subbasin number (none)
----	----	------------------------

## 5.15 bmp\_ri\_pond.f90 File Reference

### Functions/Subroutines

- subroutine [bmp\\_ri\\_pond](#) (sb, kk, rifl, rised)

*this subroutine routes water through a retention irrigation pond in the subbasin param[in] sb subbasin or reach number  
param[in] kk pond id number in the subbasin param[inout] rifl stormwater runoff coming in/out of pond at a time step  
param[inout] rised overland flow sediment coming in/out of pond at a time step*

### 5.15.1 Detailed Description

file containing the subroutine bmp\_ri\_pond

## Author

modified by Javier Burguete

## 5.16 bmp\_sand\_filter.f90 File Reference

### Functions/Subroutines

- subroutine [bmp\\_sand\\_filter](#) (sb, kk, flw, sed)

*this subroutine routes water and sediment through sand filters in the subbasin param[in] sb subbasin or reach number  
param[in] kk filter id number in the subbasin param[inout] flw stormwater runoff coming in/out of pond at a time step  
param[inout] sed overland flow sediment coming in/out of pond at a time step*

### 5.16.1 Detailed Description

file containing the subroutine `bmp_sand_filter`

Author

modified by Javier Burguete

## 5.17 `bmp_sed_pond.f90` File Reference

### Functions/Subroutines

- subroutine `bmp_sed_pond` (sb, kk, flw, sed)  
*this subroutine routes water and sediment through a sedimentation pond in the subbasin param[in] sb subbasin or reach number param[in] kk filter id number in the subbasin param[inout] flw stormwater runoff coming in/out of pond at a time step param[inout] sed overland flow sediment coming in/out of pond at a time step*

### 5.17.1 Detailed Description

file containing the subroutine `bmp_sed_pond`

Author

modified by Javier Burguete

## 5.18 `bmp_wet_pond.f90` File Reference

### Functions/Subroutines

- subroutine `bmp_wet_pond` (sb)  
*run wet pond processes*
- real \*8 function `ext_dpth` (sb)
- real \*8 function `wpnd_depth` (hvol, width, slp, lenwidth)  
*calculate ponding depth using Newton's method*
- real \*8 function `pipe_discharge` (pdia, plen, hdep, mann, mloss)  
*calculate discharge from extended detention through pvc pipe, m3/s*

### 5.18.1 Detailed Description

file containing the subroutine `bmp_wet_pond` and the functions `ext_dpth`, `wpnd_depth` and `pipe_discharge`

Author

modified by Javier Burguete

### 5.18.2 Function/Subroutine Documentation

#### 5.18.2.1 `bmp_wet_pond()`

```
subroutine bmp_wet_pond (
    integer, intent(in) sb )
```

run wet pond processes

## Parameters

in	sb	subbasin number (none)
----	----	------------------------

## 5.18.2.2 pipe\_discharge()

```
real*8 function pipe_discharge (
    real*8, intent(in) pdia,
    real*8, intent(in) plen,
    real*8, intent(in) hdep,
    real*8, intent(in) mann,
    real*8, intent(in) mloss )
```

calculate discharge from extended detention through pvc pipe,m3/s

## Parameters

out	discharge	(m <sup>3</sup> /s)
-----	-----------	---------------------

## 5.19 bmpinit.f90 File Reference

## Functions/Subroutines

- subroutine [bmpinit](#) (ii)  
*this subroutine sets default values for urban bmp parameters*

## 5.19.1 Detailed Description

file containing the subroutine bmpinit

## Author

modified by Javier Burguete

## 5.19.2 Function/Subroutine Documentation

## 5.19.2.1 bmpinit()

```
subroutine bmpinit (
    integer, intent(in) ii )
```

this subroutine sets default values for urban bmp parameters

## Parameters

<code>in</code>	<code>ii</code>	subbasin number
-----------------	-----------------	-----------------

## 5.20 buffer.f90 File Reference

### Functions/Subroutines

- subroutine `buffer` (`j`)  
*this subroutine calculates the reduction of nitrates through a riparian buffer system - developed for Sushama at NC State*

#### 5.20.1 Detailed Description

file containing the subroutine `buffer`

## Author

modified by Javier Burguete

#### 5.20.2 Function/Subroutine Documentation

##### 5.20.2.1 `buffer()`

```
subroutine buffer (
    integer, intent(in) j )
```

this subroutine calculates the reduction of nitrates through a riparian buffer system - developed for Sushama at NC State

## Parameters

<code>in</code>	<code>j</code>	HRU number (none)
-----------------	----------------	-------------------

## 5.21 burnop.f90 File Reference

### Functions/Subroutines

- subroutine `burnop` (`j`)  
*this subroutine performs burning*

### 5.21.1 Detailed Description

file containing the subroutine burnop

Author

modified by Javier Burguete

### 5.21.2 Function/Subroutine Documentation

#### 5.21.2.1 burnop()

```
subroutine burnop (
    integer, intent(in) j )
```

this subroutine performs burning

Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.22 canopyint.f90 File Reference

### Functions/Subroutines

- subroutine [canopyint](#) (j)  
*this subroutine computes canopy interception of rainfall used for methods other than curve number*

#### 5.22.1 Detailed Description

file containing the subroutine canopyint

Author

modified by Javier Burguete

#### 5.22.2 Function/Subroutine Documentation

##### 5.22.2.1 canopyint()

```
subroutine canopyint (
    integer, intent(in) j )
```

this subroutine computes canopy interception of rainfall used for methods other than curve number

## Parameters

<code>i</code>	<code>j</code>	HRU number (none)
----------------	----------------	-------------------

## 5.23 caps.f90 File Reference

### Functions/Subroutines

- subroutine `caps` (`file_name`)

*this subroutine reads the input and output names given in file.cio and converts all capital letters to lowercase letters.*

#### 5.23.1 Detailed Description

file containing the subroutine caps

#### Author

modified by Javier Burguete

#### 5.23.2 Function/Subroutine Documentation

##### 5.23.2.1 caps()

```
subroutine caps (
    character (len=*) file_name )
```

this subroutine reads the input and output names given in file.cio and converts all capital letters to lowercase letters.

## Parameters

<code>file_name</code>	dummy argument, file name character string
------------------------	--

## 5.24 carbon\_new.f90 File Reference

### Functions/Subroutines

- subroutine `carbon` (`i`, `j`)

*This code simulates organic C, N, and P cycling in the soil. It has been adapted from [2]. and crafted to accomodate to SWAT conventions. Plant residues and manure residues are decomposed separately. For convenience, the denitrification subroutine is called from here. March 2009: testing has been minimal and further adjustments are expected. Manuscript describing this subroutine to be submitted to Ecological Modelling (September, 2010). Use with caution and report anomalous results to [akemanian@psu.edu](mailto:akemanian@psu.edu), [jeff.arnold@ars.usda.edu](mailto:jeff.arnold@ars.usda.edu) and [stefan.julich@tudor.lu](mailto:stefan.julich@tudor.lu).*



- real \*8 function **fwf** (fc, wc, pwp)
- real \*8 function **fof** (void, por)
- real \*8 function **ftilf** (tillage, wc, sat)
- real \*8 function **fcx** (pclay)
- real \*8 function **fsol\_cdec** (pcarbon, cx, cfdec, tilf, csf, sol\_cmass)
- real \*8 function **fcnnew** (yy1, yy2, CNpool, yy5)
- real \*8 function **fhc** (pclay, pcarbon, cx)
- real \*8 function **fnetmin** (poold, R1, R2, hc, dummy, poolm, xinorg, cc1)

### 5.24.1 Detailed Description

file containing the subroutine carbon

#### Author

Armen R. Kemanian,  
Stefan Julich,  
modified by Javier Burguete

### 5.24.2 Function/Subroutine Documentation

#### 5.24.2.1 carbon()

```
subroutine carbon (
    integer, intent(in) i,
    integer, intent(in) j )
```

This code simulates organic C, N, and P cycling in the soil. It has been adapted from [2]. and crafted to accomodate to SWAT conventions. Plant residues and manure residues are decomposed separately. For convenience, the denitrification subroutine is called from here. March 2009: testing has been minimal and further adjustments are expected. Manuscript describing this subroutine to be submitted to Ecological Modelling (September, 2010). Use with caution and report anomalous results to [akemanian@psu.edu](mailto:akemanian@psu.edu), [jeff.arnold@ars.usda.edu](mailto:jeff.arnold@ars.usda.edu) and [stefan.julich@tudor.lu](mailto:stefan.julich@tudor.lu).

#### Parameters

<i>i</i>	current day in simulation—loop counter (julian date)
<i>j</i>	HRU number

## 5.25 carbon\_zhang2.f90 File Reference

### Functions/Subroutines

- subroutine [carbon\\_zhang2](#) (j)

### 5.25.1 Detailed Description

file containing the subroutine carbon\_zhang2

Author

modified by Javier Burguete

### 5.25.2 Function/Subroutine Documentation

#### 5.25.2.1 carbon\_zhang2()

```
subroutine carbon_zhang2 (  
    integer, intent(in) j )
```

Parameters

<i>j</i>	HRU number
----------	------------

## 5.26 cfactor.f90 File Reference

### Functions/Subroutines

- subroutine [cfactor](#) (*j*)  
*this subroutine predicts daily soil loss caused by water erosion using the modified universal soil loss equation*

#### 5.26.1 Detailed Description

file containing the subroutine cfactor

Author

modified by Javier Burguete

#### 5.26.2 Function/Subroutine Documentation

##### 5.26.2.1 cfactor()

```
subroutine cfactor (  
    integer, intent(in) j )
```

this subroutine predicts daily soil loss caused by water erosion using the modified universal soil loss equation

## Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.27 clgen.f90 File Reference

### Functions/Subroutines

- subroutine `clgen` (j)  
*this subroutine calculates the daylength, distribution of radiation throughout the day and maximum radiation for day*

#### 5.27.1 Detailed Description

file containing the subroutine `clgen`

## Author

modified by Javier Burguete

#### 5.27.2 Function/Subroutine Documentation

##### 5.27.2.1 `clgen()`

```
subroutine clgen (
    integer, intent(in) j )
```

this subroutine calculates the daylength, distribution of radiation throughout the day and maximum radiation for day

## Parameters

in	j	HRU number
----	---	------------

## 5.28 clicon.f90 File Reference

### Functions/Subroutines

- subroutine `clicon` (i)  
*this subroutine controls weather inputs to SWAT. Precipitation and temperature data is read in and the weather generator is called to fill in radiation, wind speed and relative humidity as well as missing precipitation and temperatures. Adjustments for climate changes studies are also made in this subroutine.*

### 5.28.1 Detailed Description

file containing the subroutine clicon

Author

modified by Javier Burguete

### 5.28.2 Function/Subroutine Documentation

#### 5.28.2.1 clicon()

```
subroutine clicon (
    integer, intent(in) i )
```

this subroutine controls weather inputs to SWAT. Precipitation and temperature data is read in and the weather generator is called to fill in radiation, wind speed and relative humidity as well as missing precipitation and temperatures. Adjustments for climate changes studies are also made in this subroutine.

Parameters

in	i	current day of simulation (julian date)
----	---	---

## 5.29 command.f90 File Reference

### Functions/Subroutines

- subroutine [command](#) (i)

*for every day of simulation, this subroutine steps through the command lines in the watershed configuration (.fig) file. Depending on the command code on the .fig file line, a command loop is accessed*

#### 5.29.1 Detailed Description

file containing the subroutine command

Author

modified by Javier Burguete

#### 5.29.2 Function/Subroutine Documentation

### 5.29.2.1 command()

```
subroutine command (
    integer, intent(in) i )
```

for every day of simulation, this subroutine steps through the command lines in the watershed configuration (.fig) file. Depending on the command code on the .fig file line, a command loop is accessed

#### Parameters

in	<i>i</i>	current day in simulation—loop counter (julian date)
----	----------	--

## 5.30 conapply.f90 File Reference

### Functions/Subroutines

- subroutine [conapply](#) (*j*)  
*this subroutine applies continuous pesticide*

### 5.30.1 Detailed Description

file containing the subroutine conapply

#### Author

modified by Javier Burguete

### 5.30.2 Function/Subroutine Documentation

#### 5.30.2.1 conapply()

```
subroutine conapply (
    integer, intent(in) j )
```

this subroutine applies continuous pesticide

#### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.31 confert.f90 File Reference

### Functions/Subroutines

- subroutine [confert](#) (j)  
*this subroutine simulates a continuous fertilizer operation*

#### 5.31.1 Detailed Description

file containing the subroutine confert

##### Author

modified by Javier Burguete

#### 5.31.2 Function/Subroutine Documentation

##### 5.31.2.1 confert()

```
subroutine confert (
    integer, intent(in) j )
```

this subroutine simulates a continuous fertilizer operation

##### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.32 crackflow.f90 File Reference

### Functions/Subroutines

- subroutine [crackflow](#) (j)  
*this subroutine modifies surface runoff to account for crack flow*

#### 5.32.1 Detailed Description

file containing the subroutine crackflow

##### Author

modified by Javier Burguete

## 5.32.2 Function/Subroutine Documentation

### 5.32.2.1 crackflow()

```
subroutine crackflow (  
    integer, intent(in) j )
```

this subroutine modifies surface runoff to account for crack flow

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.33 crackvol.f90 File Reference

### Functions/Subroutines

- subroutine [crackvol](#) (j)

*this subroutine computes total crack volume for the soil profile and modifies surface runoff to account for crack flow*

### 5.33.1 Detailed Description

file containing the subroutine crackvol

#### Author

modified by Javier Burguete

## 5.33.2 Function/Subroutine Documentation

### 5.33.2.1 crackvol()

```
subroutine crackvol (  
    integer, intent(in) j )
```

this subroutine computes total crack volume for the soil profile and modifies surface runoff to account for crack flow

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.34 curno.f90 File Reference

### Functions/Subroutines

- subroutine [curno](#) (cnn, h)

*this subroutine determines the curve numbers for moisture conditions I and III and calculates coefficients and shape parameters for the water retention curve. The coefficients and shape parameters are calculated by one of two methods:*

*the default method is to make them a function of soil water,*

*the alternative method (labeled new) is to make them a function of accumulated PET, precipitation and surface runoff*

### 5.34.1 Detailed Description

file containing the subroutine curno

#### Author

modified by Javier Burguete

### 5.34.2 Function/Subroutine Documentation

#### 5.34.2.1 curno()

```
subroutine curno (
    real*8, intent(in) cnn,
    integer, intent(in) h )
```

this subroutine determines the curve numbers for moisture conditions I and III and calculates coefficients and shape parameters for the water retention curve. The coefficients and shape parameters are calculated by one of two methods:

the default method is to make them a function of soil water,

the alternative method (labeled new) is to make them a function of accumulated PET, precipitation and surface runoff

#### Parameters

in	<i>cnn</i>	SCS runoff curve number for moisture condition II
in	<i>h</i>	HRU number

## 5.35 dailycn.f90 File Reference

### Functions/Subroutines

- subroutine [dailycn](#) (j)

*calculates curve number for the day in the HRU*



### 5.35.1 Detailed Description

file containing the subroutine `dailycn`

Author

modified by Javier Burguete

### 5.35.2 Function/Subroutine Documentation

#### 5.35.2.1 `dailycn()`

```
subroutine dailycn (
    integer, intent(in) j )
```

calculates curve number for the day in the HRU

Parameters

in	<i>j</i>	HRU number (none)
----	----------	-------------------

## 5.36 decay.f90 File Reference

### Functions/Subroutines

- subroutine [decay](#) (*j*)  
*this subroutine calculates degradation of pesticide in the soil and on the plants*

#### 5.36.1 Detailed Description

file containing the subroutine `decay`

Author

modified by Javier Burguete

#### 5.36.2 Function/Subroutine Documentation

##### 5.36.2.1 `decay()`

```
subroutine decay (
    integer, intent(in) j )
```

this subroutine calculates degradation of pesticide in the soil and on the plants

## Parameters

<code>in</code>	<code>j</code>	HRU number
-----------------	----------------	------------

## 5.37 depstor.f90 File Reference

### Functions/Subroutines

- subroutine [depstor](#) (`j`)  
*this subroutine computes maximum surface depressional storage depth based on random and oriented roughness and slope steepness*

#### 5.37.1 Detailed Description

file containing the subroutine depstor

#### Author

modified by Javier Burguete

#### 5.37.2 Function/Subroutine Documentation

##### 5.37.2.1 depstor()

```
subroutine depstor (
    integer, intent(in) j )
```

this subroutine computes maximum surface depressional storage depth based on random and oriented roughness and slope steepness

## Parameters

<code>in</code>	<code>j</code>	HRU number
-----------------	----------------	------------

## 5.38 distributed\_bmps.f90 File Reference

### Functions/Subroutines

- subroutine [distributed\\_bmps](#) (`sb`)  
*this subroutine calls routines for urban BMPs in the subbasin param[in] sb subbasin or reach number*

### 5.38.1 Detailed Description

file containing the subroutine distributed\_bmps

Author

modified by Javier Burguete

## 5.39 dormant.f90 File Reference

### Functions/Subroutines

- subroutine [dormant](#) (j)  
*this subroutine checks the dormant status of the different plant types*

### 5.39.1 Detailed Description

file containing the subroutine dormant

Author

modified by Javier Burguete

### 5.39.2 Function/Subroutine Documentation

#### 5.39.2.1 dormant()

```
subroutine dormant (
    integer, intent(in) j )
```

this subroutine checks the dormant status of the different plant types

Parameters

in	j	HRU number
----	---	------------

## 5.40 drains.f90 File Reference

### Functions/Subroutines

- subroutine [drains](#) (j)  
*this subroutine finds the effective lateral hydraulic conductivity and computes drainage or subirrigation flux*

### 5.40.1 Detailed Description

file containing the subroutine drains

Author

modified by Javier Burguete

### 5.40.2 Function/Subroutine Documentation

#### 5.40.2.1 drains()

```
subroutine drains (
    integer, intent(in) j )
```

this subroutine finds the effective lateral hydraulic conductivity and computes drainage or subirrigation flux

Parameters

in	j	HRU number
----	---	------------

## 5.41 dstn1.f90 File Reference

### Functions/Subroutines

- real \*8 function [dstn1](#) (rn1, rn2)

*this function computes the distance from the mean of a normal distribution with mean = 0 and standard deviation = 1, given two random numbers*

#### 5.41.1 Detailed Description

file containing the function dstn1

Author

modified by Javier Burguete

#### 5.41.2 Function/Subroutine Documentation

### 5.41.2.1 dstn1()

```
real*8 function dstn1 (  
    real*8, intent(in)  rn1,  
    real*8, intent(in)  rn2 )
```

this function computes the distance from the mean of a normal distribution with mean = 0 and standard deviation = 1, given two random numbers

**Parameters**

in	<i>rn1</i>	first random number
in	<i>rn2</i>	second random number

**Returns**

distance from the mean

## 5.42 ee.f90 File Reference

**Functions/Subroutines**

- real \*8 function `ee` (*tk*)  
*this function calculates saturation vapor pressure at a given air temperature*

### 5.42.1 Detailed Description

file containing the function `ee`

**Author**

modified by Javier Burguete

### 5.42.2 Function/Subroutine Documentation

#### 5.42.2.1 `ee()`

```
real*8 function ee (  
    real*8, intent(in) tk )
```

this function calculates saturation vapor pressure at a given air temperature

**Parameters**

in	<i>tk</i>	mean air temperature (deg C)
----	-----------	------------------------------

**Returns**

saturation vapor pressure (kPa)

## 5.43 eiusle.f90 File Reference

**Functions/Subroutines**

- subroutine `eiusle` (j)  
*this subroutine computes the USLE erosion index (EI)*

### 5.43.1 Detailed Description

file containing the subroutine `eiusle`

**Author**

modified by Javier Burguete

## 5.44 enrsb.f90 File Reference

**Functions/Subroutines**

- subroutine `enrsb` (iwave, j)  
*this subroutine calculates the enrichment ratio for nutrient and pesticide transport with runoff*

### 5.44.1 Detailed Description

file containing the subroutine `enrsb`

**Author**

modified by Javier Burguete

### 5.44.2 Function/Subroutine Documentation

#### 5.44.2.1 `enrsb()`

```
subroutine enrsb (  
    integer, intent(in) iwave,  
    integer, intent(in) j )
```

this subroutine calculates the enrichment ratio for nutrient and pesticide transport with runoff

## Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number

## 5.45 estimate\_ksat.f90 File Reference

### Functions/Subroutines

- subroutine [estimate\\_ksat](#) (perc\_clay, esti\_ksat)

*This subroutine calculates ksat value for a soil layer given the % of clay in the soil layer.*

#### 5.45.1 Detailed Description

file containing the subroutine estimate\_ksat

#### Author

modified by Javier Burguete

#### 5.45.2 Function/Subroutine Documentation

##### 5.45.2.1 estimate\_ksat()

```
subroutine estimate_ksat (
    real*8, intent(in) perc_clay,
    real*8, intent(out) esti_ksat )
```

This subroutine calculates ksat value for a soil layer given the % of clay in the soil layer.

Background: published work of Walter Rawls. Calculated ksat values based on soil texture (sand, silt and clay).  
Idea: there exists a relationship between % clay and Ksat. Equations used in this subroutine are based on the above idea (Jimmy Willimas)

## Parameters

in	<i>perc_clay</i>	clay percentage (%)
out	<i>esti_ksat</i>	estimated ksat



## 5.46 etact.f90 File Reference

### Functions/Subroutines

- subroutine `etact` (j)

*this subroutine calculates potential plant transpiration for Priestley- Taylor and Hargreaves ET methods, and potential and actual soil evaporation. NO3 movement into surface soil layer due to evaporation is also calculated.*

#### 5.46.1 Detailed Description

file containing the subroutine `etact`

Author

modified by Javier Burguete

## 5.47 etpot.f90 File Reference

### Functions/Subroutines

- subroutine `etpot` (j)

*this subroutine calculates potential evapotranspiration using one of three methods. If Penman-Monteith is being used, potential plant transpiration is also calculated.*

#### 5.47.1 Detailed Description

file containing the subroutine `etpot`

Author

modified by Javier Burguete

#### 5.47.2 Function/Subroutine Documentation

##### 5.47.2.1 `etpot()`

```
subroutine etpot (  
    integer, intent(in) j )
```

this subroutine calculates potential evapotranspiration using one of three methods. If Penman-Monteith is being used, potential plant transpiration is also calculated.

## Parameters

in	j	HRU number
----	---	------------

## 5.48 expo.f90 File Reference

### Functions/Subroutines

- real \*8 function `expo` (xx)

*this function checks the argument against upper and lower boundary values prior to taking the Exponential*

#### 5.48.1 Detailed Description

file containing the function expo

#### Author

modified by Javier Burguete

#### 5.48.2 Function/Subroutine Documentation

##### 5.48.2.1 `expo()`

```
real*8 function expo (
    real*8, intent(in) xx )
```

this function checks the argument against upper and lower boundary values prior to taking the Exponential

## Parameters

in	xx	exponential argument (none)
----	----	-----------------------------

## Returns

$\exp(xx)$

## 5.49 fcgd.f90 File Reference

### Functions/Subroutines

- real \*8 function `fcgd` (xx)

### 5.49.1 Detailed Description

file containing the function fcgd

Author

modified by Javier Burguete

## 5.50 fert.f90 File Reference

### Functions/Subroutines

- subroutine `fert` (*j*, *ifrt*)  
*this subroutine applies N and P specified by date and amount in the management file (.mgt)*

### 5.50.1 Detailed Description

file containing the subroutine fert

Author

modified by Javier Burguete

### 5.50.2 Function/Subroutine Documentation

#### 5.50.2.1 fert()

```
subroutine fert (
    integer, intent(in) j,
    integer, intent(in) ifrt )
```

this subroutine applies N and P specified by date and amount in the management file (.mgt)

Parameters

<i>in</i>	<i>j</i>	HRU number
-----------	----------	------------

## 5.51 filter.f90 File Reference

### Functions/Subroutines

- subroutine `filter` (*i*, *j*)  
*this subroutine calculates the reduction of pollutants in surface runoff due to an edge of field filter or buffer strip*

### 5.51.1 Detailed Description

file containing the subroutine filter

Author

modified by Javier Burguete

### 5.51.2 Function/Subroutine Documentation

#### 5.51.2.1 filter()

```
subroutine filter (
    integer, intent(in) i,
    integer, intent(in) j )
```

this subroutine calculates the reduction of pollutants in surface runoff due to an edge of field filter or buffer strip

Parameters

in	<i>i</i>	current day in simulation-loop counter (julian date)
in	<i>j</i>	HRU number (none)

## 5.52 filtw.f90 File Reference

### Functions/Subroutines

- subroutine [filtw](#) (j)

*this subroutine calculates the reduction of pollutants in surface runoff due to an edge of field filter or buffer strip*

#### 5.52.1 Detailed Description

file containing the subroutine filtw

Author

modified by Javier Burguete

### 5.52.2 Function/Subroutine Documentation

#### 5.52.2.1 filtw()

```
subroutine filtw (
    integer, intent(in) j )
```

this subroutine calculates the reduction of pollutants in surface runoff due to an edge of field filter or buffer strip

## Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.53 finalbal.f90 File Reference

### Functions/Subroutines

- subroutine [finalbal](#)  
*this subroutine calculates final water balance for watershed*

#### 5.53.1 Detailed Description

file containing the subroutine finalbal

## Author

modified by Javier Burguete

## 5.54 gcycl.f90 File Reference

### Functions/Subroutines

- subroutine [gcycl](#)  
*This subroutine initializes the random number seeds. If the user desires a different set of random numbers for each simulation run, the random number generator is used to reset the values of the seeds.*

#### 5.54.1 Detailed Description

file containing the subroutine gcycl

## Author

modified by Javier Burguete

## 5.55 getallo.f90 File Reference

### Functions/Subroutines

- subroutine [getallo](#)  
*This subroutine calculates the number of HRUs, subbasins, etc. in the simulation. These values are used to allocate array sizes.*

### 5.55.1 Detailed Description

file containing the subroutine getallo

Author

modified by Javier Burguete

## 5.56 grass\_wway.f90 File Reference

### Functions/Subroutines

- subroutine [grass\\_wway](#) (j)  
*this subroutine controls the grass waterways*

### 5.56.1 Detailed Description

file containing the subroutine grass\_wway

Author

modified by Javier Burguete

### 5.56.2 Function/Subroutine Documentation

#### 5.56.2.1 grass\_wway()

```
subroutine grass_wway (
    integer, intent(in) j )
```

this subroutine controls the grass waterways

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.57 graze.f90 File Reference

### Functions/Subroutines

- subroutine [graze](#) (j)  
*this subroutine simulates biomass lost to grazing*

### 5.57.1 Detailed Description

file containing the subroutine graze

Author

modified by Javier Burguete

### 5.57.2 Function/Subroutine Documentation

#### 5.57.2.1 graze()

```
subroutine graze (
    integer, intent(in) j )
```

this subroutine simulates biomass lost to grazing

Parameters

in	j	HRU number
----	---	------------

## 5.58 grow.f90 File Reference

### Functions/Subroutines

- subroutine [grow](#) (j)  
*this subroutine adjusts plant biomass, leaf area index, and canopy height taking into account the effect of water, temperature and nutrient stresses on the plant*

#### 5.58.1 Detailed Description

file containing the subroutine grow

Author

modified by Javier Burguete

#### 5.58.2 Function/Subroutine Documentation

##### 5.58.2.1 grow()

```
subroutine grow (
    integer, intent(in) j )
```

this subroutine adjusts plant biomass, leaf area index, and canopy height taking into account the effect of water, temperature and nutrient stresses on the plant

## Parameters

in	j	HRU number
----	---	------------

## 5.59 gw\_no3.f90 File Reference

### Functions/Subroutines

- subroutine [gw\\_no3](#) (j)  
*this subroutine estimates groundwater contribution to streamflow*

#### 5.59.1 Detailed Description

file containing the subroutine gw\_no3

## Author

modified by Javier Burguete

#### 5.59.2 Function/Subroutine Documentation

##### 5.59.2.1 gw\_no3()

```
subroutine gw_no3 (
    integer, intent(in) j )
```

this subroutine estimates groundwater contribution to streamflow

## Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.60 gwmod.f90 File Reference

### Functions/Subroutines

- subroutine [gwmod](#) (j)  
*this subroutine estimates groundwater contribution to streamflow*



### 5.60.1 Detailed Description

file containing the subroutine gwmod

Author

modified by Javier Burguete

### 5.60.2 Function/Subroutine Documentation

#### 5.60.2.1 gwmod()

```
subroutine gwmod (  
    integer, intent(in) j )
```

this subroutine estimates groundwater contribution to streamflow

Parameters

<i>j</i>	HRU number
----------	------------

## 5.61 gwmod\_deep.f90 File Reference

### Functions/Subroutines

- subroutine [gwmod\\_deep](#) (*j*)  
*this subroutine estimates groundwater contribution to streamflow*

#### 5.61.1 Detailed Description

file containing the subroutine gwmod\_deep

Author

modified by Javier Burguete

#### 5.61.2 Function/Subroutine Documentation

##### 5.61.2.1 gwmod\_deep()

```
subroutine gwmod_deep (  
    integer, intent(in) j )
```

this subroutine estimates groundwater contribution to streamflow

## Parameters

<i>j</i>	HRU number
----------	------------

## 5.62 gwnutr.f90 File Reference

### Functions/Subroutines

- subroutine [gwnutr](#) (*j*)  
*this subroutine calculates the nitrate and soluble phosphorus loading contributed by groundwater flow*

#### 5.62.1 Detailed Description

file containing the subroutine gwnutr

## Author

modified by Javier Burguete

#### 5.62.2 Function/Subroutine Documentation

##### 5.62.2.1 gwnutr()

```
subroutine gwnutr (
    integer, intent(in) j )
```

this subroutine calculates the nitrate and soluble phosphorus loading contributed by groundwater flow

## Parameters

in	<i>j</i>	HRU number (none)
----	----------	-------------------

## 5.63 h2omgt\_init.f90 File Reference

### Functions/Subroutines

- subroutine [h2omgt\\_init](#)  
*This subroutine initializes variables related to water management (irrigation, consumptive water use, etc.)*

### 5.63.1 Detailed Description

file containing the subroutine h2omgt\_init

Author

modified by Javier Burguete

## 5.64 harvestop.f90 File Reference

### Functions/Subroutines

- subroutine [harvestop](#) (j)  
*this subroutine performs the harvest operation (no kill)*

### 5.64.1 Detailed Description

file containing the subroutine harvestop

Author

modified by Javier Burguete

### 5.64.2 Function/Subroutine Documentation

#### 5.64.2.1 harvestop()

```
subroutine harvestop (
    integer, intent(in) j )
```

this subroutine performs the harvest operation (no kill)

Parameters

in	j	HRU number
----	---	------------

## 5.65 harvkillop.f90 File Reference

### Functions/Subroutines

- subroutine [harvkillop](#) (j)  
*this subroutine performs the harvest and kill operation*

### 5.65.1 Detailed Description

file containing the subroutine harvkillop

Author

modified by Javier Burguete

### 5.65.2 Function/Subroutine Documentation

#### 5.65.2.1 harvkillop()

```
subroutine harvkillop (
    integer, intent(in) j )
```

this subroutine performs the harvest and kill operation

Parameters

in	j	HRU number
----	---	------------

## 5.66 headout.f90 File Reference

### Functions/Subroutines

- subroutine [headout](#)  
*this subroutine writes the headings to the major output files*

#### 5.66.1 Detailed Description

file containing the subroutine headout

Author

modified by Javier Burguete

## 5.67 hhnoqual.f90 File Reference

### Functions/Subroutines

- subroutine [hhnoqual](#) (jrch)  
*this subroutine performs in-stream nutrient calculations. No transformations are calculated*

### 5.67.1 Detailed Description

file containing the subroutine hhnoqual

Author

modified by Javier Burguete

### 5.67.2 Function/Subroutine Documentation

#### 5.67.2.1 hhnoqual()

```
subroutine hhnoqual (
    integer, intent(in) jrch )
```

this subroutine performs in-stream nutrient calculations. No transformations are calculated

Parameters

in	jrch	reach number (none)
----	------	---------------------

## 5.68 hhwatqual.f90 File Reference

### Functions/Subroutines

- subroutine [hhwatqual](#) (jrch)  
*this subroutine performs in-stream nutrient transformations and water quality calculations for hourly timestep*

#### 5.68.1 Detailed Description

file containing the subroutine hhwatqual

Author

modified by Javier Burguete

#### 5.68.2 Function/Subroutine Documentation

##### 5.68.2.1 hhwatqual()

```
subroutine hhwatqual (
    integer, intent(in) jrch )
```

this subroutine performs in-stream nutrient transformations and water quality calculations for hourly timestep

## Parameters

in	jrch	reach number (none)
----	------	---------------------

## 5.69 hmeas.f90 File Reference

### Functions/Subroutines

- subroutine [hmeas](#)

*this subroutine reads in relative humidity data from file and assigns the data to the HRUs*

#### 5.69.1 Detailed Description

file containing the subroutine hmeas

#### Author

modified by Javier Burguete

## 5.70 HQDAV.f90 File Reference

### Functions/Subroutines

- subroutine [hqdav](#) (A, CBW, QQ, SSS, ZCH, ZX, CHW, FPW, jrch)

*this subprogram computes flow area and depth given rate in a reach. Adopted from APEX1501 by Jaehak Jeong 2017*

#### 5.70.1 Detailed Description

file containing the subroutine HQDAV

#### Author

Jaehak Jeong,  
modified by Javier Burguete

## 5.71 hruaa.f90 File Reference

### Functions/Subroutines

- subroutine [hruaa](#) (years)

*this subroutine writes average annual HRU output to the output.hru file*

### 5.71.1 Detailed Description

file containing the subroutine hruaa

Author

modified by Javier Burguete

### 5.71.2 Function/Subroutine Documentation

#### 5.71.2.1 hruaa()

```
subroutine hruaa (
    real*8, intent(in) years )
```

this subroutine writes average annual HRU output to the output.hru file

Parameters

in	years	length of simulation (years)
----	-------	------------------------------

## 5.72 hruallo.f90 File Reference

### Functions/Subroutines

- subroutine [hruallo](#)  
*This subroutine calculates the number of management operation types, etc. used in the simulation. These values are used to allocate array sizes for processes occurring in the HRU.*

#### 5.72.1 Detailed Description

file containing the subroutine hruallo

Author

modified by Javier Burguete

## 5.73 hruday.f90 File Reference

### Functions/Subroutines

- subroutine [hruday](#) (i, j)  
*this subroutine writes daily HRU output to the output.hru file*

### 5.73.1 Detailed Description

file containing the subroutine hruday

Author

modified by Javier Burguete

### 5.73.2 Function/Subroutine Documentation

#### 5.73.2.1 hruday()

```
subroutine hruday (
    integer, intent(in) i,
    integer, intent(in) j )
```

this subroutine writes daily HRU output to the output.hru file

Parameters

in	<i>i</i>	current day in simulation-loop counter (julian date)
in	<i>j</i>	HRU number (none)

## 5.74 hrumon.f90 File Reference

### Functions/Subroutines

- subroutine [hrumon](#)  
*this subroutine writes monthly HRU output to the output.hru file*

#### 5.74.1 Detailed Description

file containing the subroutine hrumon

Author

modified by Javier Burguete

## 5.75 hrupond.f90 File Reference

### Functions/Subroutines

- subroutine [hrupond](#) (j)  
*this subroutine routes water and sediment through ponds in the HRUs*



### 5.75.1 Detailed Description

file containing the subroutine hrupond

Author

modified by Javier Burguete

### 5.75.2 Function/Subroutine Documentation

#### 5.75.2.1 hrupond()

```
subroutine hrupond (
    integer, intent(in) j )
```

this subroutine routes water and sediment through ponds in the HRUs

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.76 hrupondhr.f90 File Reference

### Functions/Subroutines

- subroutine [hrupondhr](#) (j)  
*this subroutine routes water and sediment through ponds in the HRUs in a subdaily time step*

#### 5.76.1 Detailed Description

file containing the subroutine hrupondhr

Author

modified by Javier Burguete

### 5.76.2 Function/Subroutine Documentation

#### 5.76.2.1 hrupondhr()

```
subroutine hrupondhr (
    integer, intent(in) j )
```

this subroutine routes water and sediment through ponds in the HRUs in a subdaily time step

## Parameters

<code>in</code>	<code>j</code>	HRU number (none)
-----------------	----------------	-------------------

## 5.77 hruyr.f90 File Reference

### Functions/Subroutines

- subroutine [hruyr](#)  
*this subroutine writes annual HRU output to the output.hru file*

#### 5.77.1 Detailed Description

file containing the subroutine hruyr

## Author

modified by Javier Burguete

## 5.78 hydroinit.f90 File Reference

### Functions/Subroutines

- subroutine [hydroinit](#)  
*This subroutine computes variables related to the watershed hydrology: the time of concentration for the subbasins, lagged surface runoff, the coefficient for the peak runoff rate equation, and lateral flow travel time.*

#### 5.78.1 Detailed Description

file containing the subroutine hydroinit

## Author

modified by Javier Burguete

## 5.79 icl.f90 File Reference

### Functions/Subroutines

- integer function [icl](#) (id)  
*this function determines the month and day, given the julian date*

### 5.79.1 Detailed Description

file containing the function icl

Author

modified by Javier Burguete

### 5.79.2 Function/Subroutine Documentation

#### 5.79.2.1 icl()

```
integer function icl (
    integer, intent(in) id )
```

this function determines the month and day, given the julian date

Parameters

in	id	julian date
----	----	-------------

## 5.80 impnd\_init.f90 File Reference

### Functions/Subroutines

- subroutine [impnd\\_init](#)  
*this subroutine initializes variables related to impoundments (ponds, wetlands, reservoirs and potholes)*

#### 5.80.1 Detailed Description

file containing the subroutine impnd\_init

Author

modified by Javier Burguete

## 5.81 impndaa.f90 File Reference

### Functions/Subroutines

- subroutine [impndaa](#) (years)  
*this subroutine writes average annual HRU impondment output to the output.wtr file*

### 5.81.1 Detailed Description

file containing the subroutine `impndaa`

Author

modified by Javier Burguete

### 5.81.2 Function/Subroutine Documentation

#### 5.81.2.1 `impndaa()`

```
subroutine impndaa (
    real*8, intent(in) years )
```

this subroutine writes average annual HRU impondment output to the `output.wtr` file

Parameters

<code>in</code>	<code>years</code>	length of simulation (years)
-----------------	--------------------	------------------------------

## 5.82 `impndday.f90` File Reference

### Functions/Subroutines

- subroutine [impndday](#) (`j`, `sb`)  
*this subroutine writes daily HRU output to the `output.wtr` file*

#### 5.82.1 Detailed Description

file containing the subroutine `impndday`

Author

modified by Javier Burguete

### 5.82.2 Function/Subroutine Documentation

#### 5.82.2.1 `impndday()`

```
subroutine impndday (
    integer, intent(in) j,
    integer, intent(in) sb )
```

this subroutine writes daily HRU output to the `output.wtr` file

## Parameters

in	j	HRU number (none)
in, out	sb	subbasin number

## 5.83 impndmon.f90 File Reference

### Functions/Subroutines

- subroutine [impndmon](#)  
*this subroutine writes monthly HRU impoundment output to the output.wtr file*

#### 5.83.1 Detailed Description

file containing the subroutine impndmon

## Author

modified by Javier Burguete

## 5.84 impndyr.f90 File Reference

### Functions/Subroutines

- subroutine [impndyr](#)  
*this subroutine writes annual HRU impondment output to the output.wtr file*

#### 5.84.1 Detailed Description

file containing the subroutine impndyr

## Author

modified by Javier Burguete

## 5.85 irr\_rch.f90 File Reference

### Functions/Subroutines

- subroutine [irr\\_rch](#) (jrch)  
*this subroutine performs the irrigation operation when the water source is a reach*

### 5.85.1 Detailed Description

file containing the subroutine `irr_rch`

Author

modified by Javier Burguete

### 5.85.2 Function/Subroutine Documentation

#### 5.85.2.1 `irr_rch()`

```
subroutine irr_rch (
    integer, intent(in) jrch )
```

this subroutine performs the irrigation operation when the water source is a reach

Parameters

<code>in</code>	<code>jrch</code>	reach number (none)
-----------------	-------------------	---------------------

## 5.86 `irr_res.f90` File Reference

### Functions/Subroutines

- subroutine [irr\\_res](#) (`jres`)  
*this subroutine performs the irrigation operation when the water source is a reservoir*

#### 5.86.1 Detailed Description

file containing the subroutine `irr_res`

Author

modified by Javier Burguete

### 5.86.2 Function/Subroutine Documentation

#### 5.86.2.1 `irr_res()`

```
subroutine irr_res (
    integer, intent(in) jres )
```

this subroutine performs the irrigation operation when the water source is a reservoir

## Parameters

in	jres	reservoir number (none)
----	------	-------------------------

## 5.87 irrigate.f90 File Reference

### Functions/Subroutines

- subroutine [irrigate](#) (j, volmm)  
*this subroutine applies irrigation water to HRU*

#### 5.87.1 Detailed Description

file containing the subroutine irrigate

#### Author

modified by Javier Burguete

#### 5.87.2 Function/Subroutine Documentation

##### 5.87.2.1 irrigate()

```
subroutine irrigate (
    integer, intent(in) j,
    real*8, intent(in) volmm )
```

this subroutine applies irrigation water to HRU

## Parameters

in	j	HRU number (none)
in	volmm	depth irrigation water applied to HRU (mm H2O)

## 5.88 irrsb.f90 File Reference

### Functions/Subroutines

- subroutine [irrsb](#) (j)  
*this subroutine performs the irrigation operation when the source is the shallow or deep aquifer or a source outside the watershed*

### 5.88.1 Detailed Description

file containing the subroutine irrsb

Author

modified by Javier Burguete

### 5.88.2 Function/Subroutine Documentation

#### 5.88.2.1 irrsb()

```
subroutine irrsb (  
    integer, intent(in) j )
```

this subroutine performs the irrigation operation when the source is the shallow or deep aquifer or a source outside the watershed

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.89 jdt.f90 File Reference

### Functions/Subroutines

- integer function [jdt](#) (numdays, i, m)

*this function computes the julian date given the month and the day of the month*

#### 5.89.1 Detailed Description

file containing the function jdt

Author

modified by Javier Burguete

#### 5.89.2 Function/Subroutine Documentation



### 5.89.2.1 jdt()

```
integer function jdt (  
    integer, dimension (13), intent(in) numdays,  
    integer, intent(in) i,  
    integer, intent(in) m )
```

this function computes the julian date given the month and the day of the month

**Parameters**

in	<i>numdays</i>	julian date for last day of preceding month (where the array location is the number of the month). The dates are for leap years (numdays=ndays) (julian date)
in	<i>i</i>	day
in	<i>m</i>	month

## 5.90 killop.f90 File Reference

### Functions/Subroutines

- subroutine [killop](#) (j)  
*this subroutine performs the kill operation*

#### 5.90.1 Detailed Description

file containing the subroutine killop

**Author**

modified by Javier Burguete

#### 5.90.2 Function/Subroutine Documentation

##### 5.90.2.1 killop()

```
subroutine killop (
    integer, intent(in) j )
```

this subroutine performs the kill operation

**Parameters**

in	<i>j</i>	HRU number
----	----------	------------

## 5.91 lakeq.f90 File Reference

### Functions/Subroutines

- subroutine [lakeq](#) (jres)  
*this subroutine computes the lake hydrologic pesticide balance.*

### 5.91.1 Detailed Description

file containing the subroutine lakeq

Author

modified by Javier Burguete

### 5.91.2 Function/Subroutine Documentation

#### 5.91.2.1 lakeq()

```
subroutine lakeq (
    integer, intent(in) jres )
```

this subroutine computes the lake hydrologic pesticide balance.

Parameters

in	jres	reservoir number (none)
----	------	-------------------------

## 5.92 latsed.f90 File Reference

### Functions/Subroutines

- subroutine [latsed](#) (j)  
*this subroutine calculates the sediment load contributed in lateral flow*

#### 5.92.1 Detailed Description

file containing the subroutine latsed

Author

modified by Javier Burguete

### 5.92.2 Function/Subroutine Documentation

#### 5.92.2.1 latsed()

```
subroutine latsed (
    integer, intent(in) j )
```

this subroutine calculates the sediment load contributed in lateral flow

## Parameters

<code>in</code>	<code>j</code>	HRU number (none)
-----------------	----------------	-------------------

## 5.93 layersplit.f90 File Reference

### Functions/Subroutines

- subroutine **layersplit** (dep\_new, k)

#### 5.93.1 Detailed Description

file containing the subroutine layersplit

#### Author

modified by Javier Burguete

## 5.94 lid\_cistern.f90 File Reference

### Functions/Subroutines

- subroutine **lid\_cistern** (sb, j, k, lid\_prec)  
*simulate cistern processes*

#### 5.94.1 Detailed Description

file containing the subroutine lid\_cistern

#### Author

modified by Javier Burguete

### 5.94.2 Function/Subroutine Documentation

#### 5.94.2.1 lid\_cistern()

```
subroutine lid_cistern (
    integer, intent(in) sb,
    integer, intent(in) j,
    integer, intent(in) k,
    real*8, intent(in) lid_prec )
```

simulate cistern processes

## Parameters

in	<i>sb</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)
in	<i>k</i>	subdaily time index (none)
in	<i>lid_prec</i>	precipitation depth a LID receives in a simulation time interval (mm)

## 5.95 lid\_greenroof.f90 File Reference

### Functions/Subroutines

- subroutine [lid\\_greenroof](#) (*sb*, *j*, *k*, *lid\_prec*)  
*simulate green roof processes*

#### 5.95.1 Detailed Description

file containing the subroutine lid\_greenroof

#### Author

modified by Javier Burguete

#### 5.95.2 Function/Subroutine Documentation

##### 5.95.2.1 lid\_greenroof()

```
subroutine lid_greenroof (
    integer, intent(in) sb,
    integer, intent(in) j,
    integer, intent(in) k,
    real*8, intent(in) lid_prec )
```

simulate green roof processes

## Parameters

in	<i>sb</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)
in	<i>k</i>	subdaily time index (none)
in	<i>lid_prec</i>	precipitation depth a LID receives in a simulation time interval (mm)

## 5.96 lid\_porpavement.f90 File Reference

### Functions/Subroutines

- subroutine [lid\\_porpavement](#) (sb, j, k, lid\_prec)  
*simulate porous pavement processes*

#### 5.96.1 Detailed Description

file containing the subroutine lid\_porpavement

#### Author

modified by Javier Burguete

#### 5.96.2 Function/Subroutine Documentation

##### 5.96.2.1 lid\_porpavement()

```
subroutine lid_porpavement (
    integer, intent(in) sb,
    integer, intent(in) j,
    integer, intent(in) k,
    real*8, intent(in) lid_prec )
```

simulate porous pavement processes

#### Parameters

in	<i>sb</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)
in	<i>k</i>	subdaily time index (none)
in	<i>lid_prec</i>	precipitation depth a LID receives in a simulation time interval (mm)

## 5.97 lid\_raingarden.f90 File Reference

### Functions/Subroutines

- subroutine [lid\\_raingarden](#) (sb, j, k, lid\_prec)  
*simulate rain garden processes*

### 5.97.1 Detailed Description

file containing the subroutine lid\_raingarden

Author

modified by Javier Burguete

### 5.97.2 Function/Subroutine Documentation

#### 5.97.2.1 lid\_raingarden()

```
subroutine lid_raingarden (
    integer, intent(in) sb,
    integer, intent(in) j,
    integer, intent(in) k,
    real*8, intent(in) lid_prec )
```

simulate rain garden processes

Parameters

in	<i>sb</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)
in	<i>k</i>	subdaily time index (none)
in	<i>lid_prec</i>	precipitation depth a LID receives in a simulation time interval (mm)

## 5.98 lidinit.f90 File Reference

### Functions/Subroutines

- subroutine [lidinit](#) (i)  
*this subroutine sets default values for LID parameters*

#### 5.98.1 Detailed Description

file containing the subroutine lidinit

Author

modified by Javier Burguete

## 5.98.2 Function/Subroutine Documentation

### 5.98.2.1 lidinit()

```
subroutine lidinit (
    integer, intent(in) i )
```

this subroutine sets default values for LID parameters

#### Parameters

in	i	subbasin number
----	---	-----------------

## 5.99 lids.f90 File Reference

### Functions/Subroutines

- subroutine [lids](#) (sb, j, k, lid\_prec)  
*call subroutines to simulate green roof, rain garden, cistern and porous pavement processes*

### 5.99.1 Detailed Description

file containing the subroutine lids

#### Author

modified by Javier Burguete

## 5.99.2 Function/Subroutine Documentation

### 5.99.2.1 lids()

```
subroutine lids (
    integer, intent(in) sb,
    integer, intent(in) j,
    integer, intent(in) k,
    real*8, intent(in) lid_prec )
```

call subroutines to simulate green roof, rain garden, cistern and porous pavement processes



## Parameters

in	<i>sb</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)
in	<i>k</i>	subdaily time index (none)
in	<i>lid_prec</i>	precipitation depth a LID receives in a simulation time interval (mm)

## 5.100 log\_normal.f90 File Reference

### Functions/Subroutines

- real \*8 function [log\\_normal](#) (mu, sig)

*this function generates a random number from a lognormal distribution curve for estimating constituent concentration in the effluent of urban bmps given mean and standard deviation values. Jaehak Jeong, 2017*

#### 5.100.1 Detailed Description

file containing the function log\_normal

#### Author

modified by Javier Burguete

#### 5.100.2 Function/Subroutine Documentation

##### 5.100.2.1 log\_normal()

```
real*8 function log_normal (
    real*8, intent(in) mu,
    real*8, intent(in) sig )
```

this function generates a random number from a lognormal distribution curve for estimating constituent concentration in the effluent of urban bmps given mean and standard deviation values. Jaehak Jeong, 2017

## Parameters

in	<i>mu</i>	mean value
in	<i>standard</i>	deviation

#### Returns

value generated for distribution

## 5.101 lwqdef.f90 File Reference

### Functions/Subroutines

- subroutine [lwqdef](#) (ii)

*this subroutine assigns default values for the lake water quality (.lwq) when the lake water quality file does not exists*

#### 5.101.1 Detailed Description

file containing the subroutine lwqdef

##### Author

modified by Javier Burguete

#### 5.101.2 Function/Subroutine Documentation

##### 5.101.2.1 lwqdef()

```
subroutine lwqdef (
    integer, intent(in) ii )
```

this subroutine assigns default values for the lake water quality (.lwq) when the lake water quality file does not exists

##### Parameters

in	ii	reservoir number (none)
----	----	-------------------------

## 5.102 main.f90 File Reference

### Functions/Subroutines

- program [main](#)

*this is the main program that reads input, calls the main simulation model, and writes output*

#### 5.102.1 Detailed Description

file containing the main program that reads input, calls the main simulation model, and writes output.

##### Author

modified by Javier Burguete Tolosa

### 5.103 modparm.f90 File Reference

## Modules

- module **parm**  
*main module containing the global variables*

## Variables

- integer, parameter **parm::mvaro** = 33  
*max number of variables routed through the reach*
- integer, parameter **parm::mhruo** = 79  
*maximum number of variables written to HRU output file (output.hru) (none)*
- integer, parameter **parm::mrcho** = 62  
*maximum number of variables written to reach output file (.rch) (none)*
- integer, parameter **parm::msubo** = 24  
*maximum number of variables written to subbasin output file (output.sub) (none)*
- integer, parameter **parm::mstdo** = 113  
*max number of variables summarized in output.std*
- integer, parameter **parm::motot** = 600
- character(len=80), parameter **parm::prog** = "SWAT Sep 7 VER 2018/Rev 670"  
*SWAT program header string (name and version)*
- character(len=13), dimension(mhruo), parameter **parm::heds** = ("PRECIPmm","SNOFALLmm","SNOMELTmm","IRRmm","PETmm","ETmm","SW\_INITmm","SW\_ENDmm","PERCmm","GW\_RCHGmm","DA\_RCHGmm","REVAPmm","SA\_IRRmm","DA\_IRRmm","SA\_STmm","DA\_STmm","SURQ\_GE←Nmm","SURQ\_CNTmm","TLOSSmm","LATQGENmm","GW\_Qmm","WYLDmm","DAILYCN","TMP←\_AVdgC","TMP\_MXdgC","TMP\_MNdgc","SOL\_TMPdgc","SOLARMJ/m<sup>2</sup>","SYLdt/ha","USLEt/ha","N←APPkg/ha","P\_APPkg/ha","NAUTOk/ha","PAUTOkg/ha","NGRZkg/ha","PGRZkg/ha","NCFRTkg/ha","P←CFRTkg/ha","NRAINkg/ha","NFIXkg/ha","F-MNkg/ha","A-MNkg/ha","A-SNkg/ha","F-MPkg/ha","AO-L←Pkg/ha","L-APkg/ha","A-SPkg/ha","DNITkg/ha","NUPkg/ha","PUPkg/ha","ORGNkg/ha","ORGPkg/ha","SEDPkg/ha","NSURQkg/ha","NLATQkg/ha","NO3Lkg/ha","NO3GWkg/ha","SOLPkg/ha","P\_GWkg/ha","W\_STRS","TMP\_STRS","N\_STRS","P\_STRS","BIOMt/ha","LAI","YLDT/ha","BACTPct ","BACTL←Pct","WTAB CLIm","WTAB SOLm","SNOmm","CMUPkg/ha","CMTOTkg/ha","QTILEmm","TNO3kg/ha","LNO3kg/ha","GW\_Q\_Dmm","LATQCNTmm","TVAPkg/ha")  
*column headers for HRU output file*
- character(len=13), dimension(msubo), parameter **parm::hedb** = ("PRECIPmm","SNOMELTmm","P←ETmm","ETmm","SWmm","PERCmm","SURQmm","GW\_Qmm","WYLDmm","SYLdt/ha","ORG←Nkg/ha","ORGPkg/ha","NSURQkg/ha","SOLPkg/ha","SEDPkg/ha","LAT Q(mm)","LATNO3kg/h","GWN←O3kg/ha","CHOLAmic/L","CBODU mg/L","DOXQ mg/L","TNO3kg/ha","QTILEmm","TVAPkg/ha/")  
*column headers for subbasin output file*
- character(len=13), dimension(mrcho), parameter **parm::heder** = ("FLOW\_INcms","FLOW\_OUTcms","EVAPcms","TLOSScms","SED\_INTons","SED\_OUTtons","SEDCONcmg/L","ORGN\_INkg","ORG←N\_OUTkg","ORGP\_INkg","ORGP\_OUTkg","NO3\_INkg","NO3\_OUTkg","NH4\_INkg","NH4\_OUTkg","NO2\_INkg","NO2\_OUTkg","MINP\_INkg","MINP\_OUTkg","CHLA\_INkg","CHLA\_OUTkg","CBOD\_INkg","CBOD\_OUTkg","DISOX\_INkg","DISOX\_OUTkg","SOLPST\_INmg","SOLPST\_OUTmg","SORPST\_I←Nmg","SORPST\_OUTmg","REACTPSTmg","VOLPSTmg","SETTLPSTmg","RESUSP\_PSTmg","DIFF←USEPSTmg","REACBEDPSTmg","BURYPSTmg","BED\_PSTmg","BACTP\_OUTct","BACTLP\_OUTct","CMETAL#1kg","CMETAL#2kg","CMETAL#3kg","TOT Nkg","TOT Pkg","NO3ConcMg/l","WTMPdegc","","","")  
*column headers for reach output file*

- character(len=13), dimension(41), parameter **parm::hedrsv** = (/ " VOLUMEm3", " FLOW\_INcms", " FLOW\_OUTcms", " PRECIPm3", " EVAPm3", " SEEPAGEm3", " SED\_INtons", " SED\_OUTtons", " SED\_CONC", " ORGN\_INkg", " ORGN\_OUTkg", " RES\_ORGNppm", " ORGP\_INkg", " ORGP\_OUTkg", " RES\_ORGPppm", " NO3\_INkg", " NO3\_OUTkg", " RES\_NO3ppm", " NO2\_INkg", " NO2\_OUTkg", " RES\_NO2ppm", " NH3\_INkg", " NH3\_OUTkg", " RES\_NH3ppm", " MINP\_INkg", " MINP\_OUTkg", " RES\_MINPppm", " CHLA\_INkg", " CHLA\_OUTkg", " SECCHIDEPTHm", " PEST\_INmg", " REACTPSTmg", " VOLPSTmg", " SETTLPS", " RESUSP\_PSTmg", " DIFFUSEPSTmg", " REACBEDPSTmg", " BURYPSTmg", " PEST\_OUTmg", " PSTCNCWmg/m3", " PSTCNCBmg/m3" /)

*column headers for reservoir output file*

- character(len=13), dimension(40), parameter **parm::hedwtr** = (/ " PNDPCPmm", " PND\_INmm", " PSED", " PNDVOLm^3", " PNDORGNppm", " PNDNO3ppm", " PNDORGPppm", " PNDMINPppm", " PNDCHLAppm", " PNDSEClm", " WETPCPmm", " WET\_INmm", " WSED", " WETEVmm", " WETSEmm", " WET\_OUTmm", " WSED", " WETVOLm^3", " WETORGNppm", " WETNO3ppm", " WETORGPppm", " WETMINPppm", " WETCHLAppm", " WETSEClm", " POTPCPmm", " POT\_INmm", " OSED", " POTEVmm", " POTSEmm", " POT\_OUTmm", " POTVOLm^3", " POT\_SAha", " HRU\_SURQmm", " PLANT\_ETmm", " SOIL\_ETmm" /)

*column headers for HRU impoundment output file*

- integer, dimension(mhruo), parameter **parm::icol** = (/43,53,63,73,83,93,103,113,123,133,143,153,163,173,183,193,203,213,223,233,243,253,263,273,283,293,303,313,323,333,343,353,363,373,383,393,403,413,423,433,443,453,463,473,483,493,503,513,523,533,543,553,563,573,583,593,603,613,623,633,643,653,663,673,683,693,703,713,723,733,743,753,763,773,783,793,803,813,823,833,843,853,863,873,883,893,903,913,923,933,943,953,963,973,983,993,1003,1013,1023,1033,1043,1053,1063,1073,1083,1093,1103,1113,1123,1133,1143,1153,1163,1173,1183,1193,1203,1213,1223,1233,1243,1253,1263,1273,1283,1293,1303,1313,1323,1333,1343,1353,1363,1373,1383,1393,1403,1413,1423,1433,1443,1453,1463,1473,1483,1493,1503,1513,1523,1533,1543,1553,1563,1573,1583,1593,1603,1613,1623,1633,1643,1653,1663,1673,1683,1693,1703,1713,1723,1733,1743,1753,1763,1773,1783,1793,1803,1813,1823,1833,1843,1853,1863,1873,1883,1893,1903,1913,1923,1933,1943,1953,1963,1973,1983,1993,2003,2013,2023,2033,2043,2053,2063,2073,2083,2093,2103,2113,2123,2133,2143,2153,2163,2173,2183,2193,2203,2213,2223,2233,2243,2253,2263,2273,2283,2293,2303,2313,2323,2333,2343,2353,2363,2373,2383,2393,2403,2413,2423,2433,2443,2453,2463,2473,2483,2493,2503,2513,2523,2533,2543,2553,2563,2573,2583,2593,2603,2613,2623,2633,2643,2653,2663,2673,2683,2693,2703,2713,2723,2733,2743,2753,2763,2773,2783,2793,2803,2813,2823,2833,2843,2853,2863,2873,2883,2893,2903,2913,2923,2933,2943,2953,2963,2973,2983,2993,3003,3013,3023,3033,3043,3053,3063,3073,3083,3093,3103,3113,3123,3133,3143,3153,3163,3173,3183,3193,3203,3213,3223,3233,3243,3253,3263,3273,3283,3293,3303,3313,3323,3333,3343,3353,3363,3373,3383,3393,3403,3413,3423,3433,3443,3453,3463,3473,3483,3493,3503,3513,3523,3533,3543,3553,3563,3573,3583,3593,3603,3613,3623,3633,3643,3653,3663,3673,3683,3693,3703,3713,3723,3733,3743,3753,3763,3773,3783,3793,3803,3813,3823,3833,3843,3853,3863,3873,3883,3893,3903,3913,3923,3933,3943,3953,3963,3973,3983,3993,4003,4013,4023,4033,4043,4053,4063,4073,4083,4093,4103,4113,4123,4133,4143,4153,4163,4173,4183,4193,4203,4213,4223,4233,4243,4253,4263,4273,4283,4293,4303,4313,4323,4333,4343,4353,4363,4373,4383,4393,4403,4413,4423,4433,4443,4453,4463,4473,4483,4493,4503,4513,4523,4533,4543,4553,4563,4573,4583,4593,4603,4613,4623,4633,4643,4653,4663,4673,4683,4693,4703,4713,4723,4733,4743,4753,4763,4773,4783,4793,4803,4813,4823,4833,4843,4853,4863,4873,4883,4893,4903,4913,4923,4933,4943,4953,4963,4973,4983,4993,5003,5013,5023,5033,5043,5053,5063,5073,5083,5093,5103,5113,5123,5133,5143,5153,5163,5173,5183,5193,5203,5213,5223,5233,5243,5253,5263,5273,5283,5293,5303,5313,5323,5333,5343,5353,5363,5373,5383,5393,5403,5413,5423,5433,5443,5453,5463,5473,5483,5493,5503,5513,5523,5533,5543,5553,5563,5573,5583,5593,5603,5613,5623,5633,5643,5653,5663,5673,5683,5693,5703,5713,5723,5733,5743,5753,5763,5773,5783,5793,5803,5813,5823,5833,5843,5853,5863,5873,5883,5893,5903,5913,5923,5933,5943,5953,5963,5973,5983,5993,6003,6013,6023,6033,6043,6053,6063,6073,6083,6093,6103,6113,6123,6133,6143,6153,6163,6173,6183,6193,6203,6213,6223,6233,6243,6253,6263,6273,6283,6293,6303,6313,6323,6333,6343,6353,6363,6373,6383,6393,6403,6413,6423,6433,6443,6453,6463,6473,6483,6493,6503,6513,6523,6533,6543,6553,6563,6573,6583,6593,6603,6613,6623,6633,6643,6653,6663,6673,6683,6693,6703,6713,6723,6733,6743,6753,6763,6773,6783,6793,6803,6813,6823,6833,6843,6853,6863,6873,6883,6893,6903,6913,6923,6933,6943,6953,6963,6973,6983,6993,7003,7013,7023,7033,7043,7053,7063,7073,7083,7093,7103,7113,7123,7133,7143,7153,7163,7173,7183,7193,7203,7213,7223,7233,7243,7253,7263,7273,7283,7293,7303,7313,7323,7333,7343,7353,7363,7373,7383,7393,7403,7413,7423,7433,7443,7453,7463,7473,7483,7493,7503,7513,7523,7533,7543,7553,7563,7573,7583,7593,7603,7613,7623,7633,7643,7653,7663,7673,7683,7693,7703,7713,7723,7733,7743,7753,7763,7773,7783,7793,7803,7813,7823,7833,7843,7853,7863,7873,7883,7893,7903,7913,7923,7933,7943,7953,7963,7973,7983,7993,8003,8013,8023,8033,8043,8053,8063,8073,8083,8093,8103,8113,8123,8133,8143,8153,8163,8173,8183,8193,8203,8213,8223,8233,8243,8253,8263,8273,8283,8293,8303,8313,8323,8333,8343,8353,8363,8373,8383,8393,8403,8413,8423,8433,8443,8453,8463,8473,8483,8493,8503,8513,8523,8533,8543,8553,8563,8573,8583,8593,8603,8613,8623,8633,8643,8653,8663,8673,8683,8693,8703,8713,8723,8733,8743,8753,8763,8773,8783,8793,8803,8813,8823,8833,8843,8853,8863,8873,8883,8893,8903,8913,8923,8933,8943,8953,8963,8973,8983,8993,9003,9013,9023,9033,9043,9053,9063,9073,9083,9093,9103,9113,9123,9133,9143,9153,9163,9173,9183,9193,9203,9213,9223,9233,9243,9253,9263,9273,9283,9293,9303,9313,9323,9333,9343,9353,9363,9373,9383,9393,9403,9413,9423,9433,9443,9453,9463,9473,9483,9493,9503,9513,9523,9533,9543,9553,9563,9573,9583,9593,9603,9613,9623,9633,9643,9653,9663,9673,9683,9693,9703,9713,9723,9733,9743,9753,9763,9773,9783,9793,9803,9813,9823,9833,9843,9853,9863,9873,9883,9893,9903,9913,9923,9933,9943,9953,9963,9973,9983,9993,1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010,1011,1012,1013,1014,1015,1016,1017,1018,1019,1020,1021,1022,1023,1024,1025,1026,1027,1028,1029,1030,1031,1032,1033,1034,1035,1036,1037,1038,1039,1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055,1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071,1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087,1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103,1104,1105,1106,1107,1108,1109,1110,1111,1112,1113,1114,1115,1116,1117,1118,1119,1120,1121,1122,1123,1124,1125,1126,1127,1128,1129,1130,1131,1132,1133,1134,1135,1136,1137,1138,1139,1140,1141,1142,1143,1144,1145,1146,1147,1148,1149,1150,1151,1152,1153,1154,1155,1156,1157,1158,1159,1160,1161,1162,1163,1164,1165,1166,1167,1168,1169,1170,1171,1172,1173,1174,1175,1176,1177,1178,1179,1180,1181,1182,1183,1184,1185,1186,1187,1188,1189,1190,1191,1192,1193,1194,1195,1196,1197,1198,1199,1200,1201,1202,1203,1204,1205,1206,1207,1208,1209,1210,1211,1212,1213,1214,1215,1216,1217,1218,1219,1220,1221,1222,1223,1224,1225,1226,1227,1228,1229,1230,1231,1232,1233,1234,1235,1236,1237,1238,1239,1240,1241,1242,1243,1244,1245,1246,1247,1248,1249,1250,1251,1252,1253,1254,1255,1256,1257,1258,1259,1260,1261,1262,1263,1264,1265,1266,1267,1268,1269,1270,1271,1272,1273,1274,1275,1276,1277,1278,1279,1280,1281,1282,1283,1284,1285,1286,1287,1288,1289,1290,1291,1292,1293,1294,1295,1296,1297,1298,1299,1300,1301,1302,1303,1304,1305,1306,1307,1308,1309,1310,1311,1312,1313,1314,1315,1316,1317,1318,1319,1320,1321,1322,1323,1324,1325,1326,1327,1328,1329,1330,1331,1332,1333,1334,1335,1336,1337,1338,1339,1340,1341,1342,1343,1344,1345,1346,1347,1348,1349,1350,1351,1352,1353,1354,1355,1356,1357,1358,1359,1360,1361,1362,1363,1364,1365,1366,1367,1368,1369,1370,1371,1372,1373,1374,1375,1376,1377,1378,1379,1380,1381,1382,1383,1384,1385,1386,1387,1388,1389,1390,1391,1392,1393,1394,1395,1396,1397,1398,1399,1400,1401,1402,1403,1404,1405,1406,1407,1408,1409,1410,1411,1412,1413,1414,1415,1416,1417,1418,1419,1420,1421,1422,1423,1424,1425,1426,1427,1428,1429,1430,1431,1432,1433,1434,1435,1436,1437,1438,1439,1440,1441,1442,1443,1444,1445,1446,1447,1448,1449,1450,1451,1452,1453,1454,1455,1456,1457,1458,1459,1460,1461,1462,1463,1464,1465,1466,1467,1468,1469,1470,1471,1472,1473,1474,1475,1476,1477,1478,1479,1480,1481,1482,1483,1484,1485,1486,1487,1488,1489,1490,1491,1492,1493,1494,1495,1496,1497,1498,1499,1500,1501,1502,1503,1504,1505,1506,1507,1508,1509,1510,1511,1512,1513,1514,1515,1516,1517,1518,1519,1520,1521,1522,1523,1524,1525,1526,1527,1528,1529,1530,1531,1532,1533,1534,1535,1536,1537,1538,1539,1540,1541,1542,1543,1544,1545,1546,1547,1548,1549,1550,1551,1552,1553,1554,1555,1556,1557,1558,1559,1560,1561,1562,1563,1564,1565,1566,1567,1568,1569,1570,1571,1572,1573,1574,1575,1576,1577,1578,1579,1580,1581,1582,1583,1584,1585,1586,1587,1588,1589,1590,1591,1592,1593,1594,1595,1596,1597,1598,1599,1600,1601,1602,1603,1604,1605,1606,1607,1608,1609,1610,1611,1612,1613,1614,1615,1616,1617,1618,1619,1620,1621,1622,1623,1624,1625,1626,1627,1628,1629,1630,1631,1632,1633,1634,1635,1636,1637,1638,1639,1640,1641,1642,1643,1644,1645,1646,1647,1648,1649,1650,1651,1652,1653,1654,1655,1656,1657,1658,1659,1660,1661,1662,1663,1664,1665,1666,1667,1668,1669,1670,1671,1672,1673,1674,1675,1676,1677,1678,1679,1680,1681,1682,1683,1684,1685,1686,1687,1688,1689,1690,1691,1692,1693,1694,1695,1696,1697,1698,1699,1700,1701,1702,1703,1704,1705,1706,1707,1708,1709,1710,1711,1712,1713,1714,1715,1716,1717,1718,1719,1720,1721,1722,1723,1724,1725,1726,1727,1728,1729,1730,1731,1732,1733,1734,1735,1736,1737,1738,1739,1740,1741,1742,1743,1744,1745,1746,1747,1748,1749,1750,1751,1752,1753,1754,1755,1756,1757,1758,1759,1760,1761,1762,1763,1764,1765,1766,1767,1768,1769,1770,1771,1772,1773,1774,1775,1776,1777,1778,1779,1780,1781,1782,1783,1784,1785,1786,1787,1788,1789,1790,1791,1792,1793,1794,1795,1796,1797,1798,1799,1800,1801,1802,1803,1804,1805,1806,1807,1808,1809,1810,1811,1812,1813,1814,1815,1816,1817,1818,1819,1820,1821,1822,1823,1824,1825,1826,1827,1828,1829,1830,1831,1832,1833,1834,1835,1836,1837,1838,1839,1840,1841,1842,1843,1844,1845,1846,1847,1848,1849,1850,1851,1852,1853,1854,1855,1856,1857,1858,1859,1860,1861,1862,1863,1864,1865,1866,1867,1868,1869,1870,1871,1872,1873,1874,1875,1876,1877,1878,1879,1880,1881,1882,1883,1884,1885,1886,1887,1888,1889,1890,1891,1892,1893,1894,1895,1896,1897,1898,1899,1900,1901,1902,1903,1904,1905,1906,1907,1908,1909,1910,1911,1912,1913,1914,1915,1916,1917,1918,1919,1920,1921,1922,1923,1924,1925,1926,1927,1928,1929,1930,1931,1932,1933,1934,1935,1936,1937,1938,1939,1940,1941,1942,1943,1944,1945,1946,1947,1948,1949,1950,1951,1952,1953,1954,1955,1956,1957,1958,1959,1960,1961,1962,1963,1964,1965,1966,1967,1968,1969,1970,1971,1972,1973,1974,1975,1976,1977,1978,1979,1980,1981,1982,1983,1984,1985,1986,1987,1988,1989,1990,1991,1992,1993,1994,1995,1996,1997,1998,1999,2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016,2017,2018,2019,2020,2021,2022,2023,2024,2025,2026,2027,2028,2029,2030,2031,2032,2033,2034,2035,2036,2037,2038,2039,2040,2041,2042,2043,2044,2045,2046,2047,2048,2049,2050,2051,2052,2053,2054,2055,2056,2057,2058,2059,2060,2061,2062,2063,2064,2065,2066,2067,2068,2069,2070,2071,2072,2073,2074,2075,2076,2077,2078,2079,2080,2081,2082,2083,2084,2085,2086,2087,2088,2089,2090,2091,2092,2093,2094,2095,2096,2097,2098,2099,2100,2101,2102,2103,2104,2105,2106,2107,2108,2109,2110,2111,2112,2113,2114,2115,2116,2117,2118,2119,2120,2121,2122,2123,2124,2125,2126,2127,2128,2129,2130,2131,2132,2133,2134,2135,2136,2137,2138,2139,2140,2141,2142,2143,2144,2145,2146,2147,2148,2149,2150,2151,2152,2153,2154,2155,2156,2157,2158,2159,2160,2161,2162,2163,2164,2165,2166,2167,2168,2169,2170,2171,2172,2173,2174,2175,2176,2177,2178,2179,2180,2181,2182,2183,2184,2185,2186,2187,2188,2189,2190,2191,2192,2193,2194,2195,2196,2197,2198,2199,2200,2201,2202,2203,2204,2205,2206,2207,2208,2209,2210,2211,2212,2213,2214,2215,2216,2217,2218,2219,2220,2221,2222,2223,2224,2225,2226,2227,2228,2229,2230,2231,2232,2233,2234,2235,2236,2237,2238,2239,2240,2241,2242,2243,2244,2245,2246,2247,2248,2249,2250,2251,2252,2253,2254,2255,2256,2257,2258,2259,2260,2261,2262,2263,2264,2265,2266,2267,2268,2269,2270,2271,2272,2273,2274,2275,2276,2277,2278,2279,2280,2281,2282,2283,2284,2285,2286,2287,2288,2289,2290,2291,2292,2293,2294,2295,2296,2297,2298,2299,2300,2301,2302,2303,2304,2305,2306,2307,2308,2309,2310,2311,2312,2313,2314,2315,2316,2317,2318,2319,2320,2321,2322,2323,2324,2325,2326,2327,2328,2329

- denitrification exponential rate coefficient*

    - real \*8, dimension(:), allocatable **parm::nperco**

*nitrate percolation coefficient (0-1)*  
*0:concentration of nitrate in surface runoff is zero*  
*1:percolate has same concentration of nitrate as surface runoff*
  - real \*8, dimension(:), allocatable **parm::surlag**

*Surface runoff lag time. This parameter is needed in subbasins where the time of concentration is greater than 1 day. SURLAG is used to create a "storage" for surface runoff to allow the runoff to take longer than 1 day to reach the subbasin outlet (days)*
  - real \*8, dimension(:), allocatable **parm::cmn**

*rate factor for humus mineralization on active organic N*
  - real \*8, dimension(:), allocatable **parm::phoskd**

*phosphorus soil partitioning coefficient. Ratio of soluble phosphorus in surface layer attached to sediment to phosphorus dissolved in soil water*
  - real \*8, dimension(:), allocatable **parm::psp**

*phosphorus availability index. The fraction of fertilizer P remaining in labile pool after initial rapid phase of P sorption (none)*
  - real \*8, dimension(:), allocatable **parm::sdnco**

*denitrification threshold: fraction of field capacity triggering denitrification*
  - real \*8 **parm::pst\_kg**

*amount of pesticide applied to HRU (kg/ha)*
  - real \*8 **parm::yield**

*yield (dry weight) (kg)*
  - real \*8 **parm::burn\_frlb**

*fraction of biomass and residue that burn(input in management file) range (0 - 1.0) (none)*
  - real \*8 **parm::yieldgrn**
  - real \*8 **parm::yieldbms**
  - real \*8 **parm::yieldtbr**
  - real \*8 **parm::yieldn**
  - real \*8 **parm::yieldp**
  - real \*8 **parm::hi\_bms**
  - real \*8 **parm::hi\_rsd**
  - real \*8 **parm::yieldrsd**
  - real \*8, dimension(:,:), allocatable **parm::hru\_rufr**
  - real \*8, dimension(:,:), allocatable **parm::daru\_km**
  - real \*8, dimension(:,:), allocatable **parm::ru\_k**
  - real \*8, dimension(:,:), allocatable **parm::ru\_c**
  - real \*8, dimension(:,:), allocatable **parm::ru\_eiq**
  - real \*8, dimension(:,:), allocatable **parm::ru\_ovsl**
  - real \*8, dimension(:,:), allocatable **parm::ru\_a**
  - real \*8, dimension(:,:), allocatable **parm::ru\_ovs**
  - real \*8, dimension(:,:), allocatable **parm::ru\_ktc**
  - real \*8, dimension(:), allocatable **parm::gwq\_ru**
  - real \*8, dimension(:), allocatable **parm::qdayout**
  - integer, dimension(:), allocatable **parm::ils2**
  - integer, dimension(:), allocatable **parm::ils2flag**
  - integer **parm::ipest**
- pesticide identification number from pest.dat (none)*
- integer **parm::iru**
  - integer **parm::mru**
  - integer **parm::irch**
  - integer **parm::isub**
  - integer **parm::mhyd\_bsn**
  - integer **parm::ils\_nofig**

- integer **parm::mhru1**
- real \*8 **parm::wshd\_sepno3**
- real \*8 **parm::wshd\_sepnh3**
- real \*8 **parm::wshd\_seporgn**
- real \*8 **parm::wshd\_sepfon**
- real \*8 **parm::wshd\_seporgp**
- real \*8 **parm::wshd\_sepfpop**
- real \*8 **parm::wshd\_sepsolp**
- real \*8 **parm::wshd\_sepbod**
- real \*8 **parm::wshd\_sepmm**
- integer, dimension(:), allocatable **parm::isep\_hru**
- real \*8 **parm::fixco**  
*nitrogen fixation coefficient*
- real \*8 **parm::nfixmx**  
*maximum daily n-fixation (kg/ha)*
- real \*8 **parm::res\_stlr\_co**  
*reservoir sediment settling coefficient*
- real \*8 **parm::rsd\_covco**  
*residue cover factor for computing fraction of cover*
- real \*8 **parm::vcrit**  
*critical velocity*
- real \*8 **parm::wshd\_snob**  
*average amount of water stored in snow at the beginning of the simulation for the entire watershed (mm H2O)*
- real \*8 **parm::wshd\_sw**  
*water in soil at beginning of simulation, or\ average amount of water stored in soil for the entire watershed, or\ difference between mass balance calculated from watershed averages and actual value for water in soil at end of simulation (goal is to have wshd\_sw = 0.) (mm H2O)*
- real \*8 **parm::wshd\_pndfr**  
*fraction of watershed area which drains into ponds (none)*
- real \*8 **parm::wshd\_pndsed**  
*total amount of suspended sediment in ponds in the watershed (metric tons),  
or mass balance discrepancy for pond sediment expressed as loading per unit hectare of drainage area (metric tons/ha)*
- real \*8 **parm::wshd\_pndv**  
*total volume of water in ponds in the watershed (m<sup>3</sup>), or mass balance discrepancy for pond water volume expressed as depth over drainage area (mm H2O)*
- real \*8 **parm::percop**  
*pesticide percolation coefficient (0-1)  
0: concentration of pesticide in surface runoff is zero  
1: percolate has same concentration of pesticide as surface runoff*
- real \*8 **parm::wshd\_resfr**  
*fraction of watershed area that drains into reservoirs (none)*
- real \*8 **parm::wshd\_pndha**  
*watershed area in hectares which drains into ponds (ha)*
- real \*8 **parm::wshd\_resha**  
*watershed area in hectares which drains into reservoirs (ha)*
- real \*8 **parm::wshd\_fminp**  
*average annual amount of mineral P applied in watershed (kg P/ha)*
- real \*8 **parm::wshd\_fnh3**  
*average annual amount of NH3-N applied in watershed (kg N/ha)*
- real \*8 **parm::wshd\_fno3**  
*average annual amount of NO3-N applied in watershed (kg N/ha)*
- real \*8 **parm::wshd\_forgn**

- average annual amount of organic N applied in watershed (kg N/ha)*

  - real \*8 `parm::wshd_ftotn`
- average annual amount of N (mineral & organic) applied in watershed (kg N/ha)*

  - real \*8 `parm::wshd_forgp`
- average annual amount of organic P applied in watershed (kg P/ha)*

  - real \*8 `parm::wshd_ftotp`
- average annual amount of P (mineral & organic) applied in watershed (kg P/ha)*

  - real \*8 `parm::wshd_yldn`
- amount of nitrogen removed from soil in watershed in the yield (kg N/ha)*

  - real \*8 `parm::wshd_yldp`
- amount of phosphorus removed from soil in watershed in the yield (kg P/ha)*

  - real \*8 `parm::wshd_fixn`
- average annual amount of nitrogen added to plant biomass via fixation (kg N/ha)*

  - real \*8 `parm::wshd_pup`
- average annual amount of plant uptake of phosphorus (kg P/ha)*

  - real \*8 `parm::wshd_nstrs`
- average annual number of nitrogen stress units in watershed (stress units)*

  - real \*8 `parm::wshd_pstrs`
- average annual number of phosphorus stress units in watershed (stress units)*

  - real \*8 `parm::wshd_tstrs`
- average annual number of temperature stress units in watershed (stress units)*

  - real \*8 `parm::wshd_wstrs`
- average annual number of water stress units in watershed (stress units)*

  - real \*8 `parm::wshd_astrs`
- initial soil water content expressed as a fraction of field capacity*

  - real \*8 `parm::ffcb`
- average annual amount of nitrogen lost from nitrate pool due to denitrification in watershed (kg N/ha)*

  - real \*8 `parm::wshd_dnit`
- average annual amount of nitrogen moving from active organic to nitrate pool in watershed (kg N/ha)*

  - real \*8 `parm::wshd_hmn`
- average annual amount of phosphorus moving from organic to labile pool in watershed (kg P/ha)*

  - real \*8 `parm::wshd_hmp`
- average annual amount of nitrogen moving from fresh organic (residue) to nitrate and active organic pools in watershed (kg N/ha)*

  - real \*8 `parm::wshd_rmn`
- average annual amount of nitrogen moving from active organic to stable organic pool in watershed (kg N/ha)*

  - real \*8 `parm::wshd_rwn`
- die-off factor for persistent bacteria in soil solution (1/day)*

  - real \*8 `parm::wdpq`
- average annual amount of phosphorus moving from fresh organic (residue) to labile and organic pools in watershed (kg P/ha)*

  - real \*8 `parm::wshd_rmp`
- average annual amount of nitrogen moving from the NH<sub>3</sub> to the NO<sub>3</sub> pool by nitrification in the watershed (kg N/ha)*

  - real \*8 `parm::wshd_nitn`
- average annual amount of nitrogen lost by ammonia volatilization in watershed (kg N/ha)*

  - real \*8 `parm::wshd_voln`
- average annual amount of phosphorus moving from labile mineral to active mineral pool in watershed (kg P/ha)*

  - real \*8 `parm::wshd_pal`
- average annual amount of phosphorus moving from active mineral to stable mineral pool in watershed (kg P/ha)*

  - real \*8 `parm::wshd_pas`
- average annual amount of phosphorus moving from active mineral to stable mineral pool in watershed (kg P/ha)*

  - real \*8 `parm::wof_p`

- fraction of persistent bacteria on foliage that is washed off by a rainfall event (none)*

  - real \*8 **parm::wshd\_raino3**  
*average annual amount of NO3 added to soil by rainfall in watershed (kg N/ha)*
  - real \*8 **parm::wshd\_plch**  
*average annual amount of phosphorus leached into second soil layer (kg P/ha)*
  - real \*8 **parm::ressedc**  
*net change in sediment in reservoir during day (metric tons)*
  - real \*8 **parm::basno3f**  
*final average amount of nitrogen in the nitrate pool in watershed soil (kg N/ha)*
  - real \*8 **parm::basorgnf**  
*final average amount of nitrogen in the organic N pool in watershed soil (kg N/ha)*
  - real \*8 **parm::wshd\_pinlet**
  - real \*8 **parm::wshd\_ptile**
  - real \*8 **parm::sftmp**  
*Snowfall temperature (deg C)*
  - real \*8 **parm::smfmx**  
*Minimum melt rate for snow during year (Dec. 21) where deg C refers to the air temperature. (mm/deg C/day)*
  - real \*8 **parm::smfmx**  
*Maximum melt rate for snow during year (June 21) where deg C refers to the air temperature. SMFMX and SMFMN allow the rate of snow melt to vary through the year. These parameters are accounting for the impact of soil temperature on snow melt. (mm/deg C/day)*
  - real \*8 **parm::smtmp**  
*Snow melt base temperature. Mean air temperature at which snow melt will occur. (deg C)*
  - real \*8 **parm::basminpf**  
*final average amount of phosphorus in the mineral P pool in watershed soil (kg P/ha)*
  - real \*8 **parm::basorgpf**  
*final average amount of phosphorus in the organic P pool in watershed soil (kg P/ha)*
  - real \*8 **parm::wshd\_ressed**  
*total amount of suspended sediment in reservoirs in the watershed (metric tons),  
or mass balance discrepancy for reservoir sediment expressed as loading per unit hectare of drainage area (metric tons/ha)*
  - real \*8 **parm::wshd\_resv**  
*total volume of water in all reservoirs in the watershed (m<sup>3</sup>),  
or mass balance discrepancy for reservoir water volume expressed as depth over drainage area (mm H2O)*
  - real \*8 **parm::basminpi**  
*average amount of phosphorus initially in the mineral P pool in watershed soil (kg P/ha)*
  - real \*8 **parm::basno3i**  
*average amount of nitrogen initially in the nitrate pool in watershed soil (kg N/ha)*
  - real \*8 **parm::basorgni**  
*average amount of nitrogen initially in the organic N pool in watershed soil (kg N/ha)*
  - real \*8 **parm::basorgpi**  
*average amount of phosphorus initially in the organic P pool in watershed soil (kg P/ha)*
  - real \*8 **parm::peakr**  
*peak runoff rate for the day in HRU or channel (m<sup>3</sup>/s)*
  - real \*8 **parm::albday**  
*albedo of ground for the day in HRU, the fraction of the solar radiation reflected at the soil surface back into space (none)*
  - real \*8 **parm::pndsedin**  
*sediment inflow to the pond from HRU during day (metric tons)*
  - real \*8 **parm::sw\_excess**  
*amount of water stored in soil layer on the current day that exceeds field capacity (gravity drained water) (mm H2O)*
  - real \*8 **parm::timp**



*Snow pack temperature lag factor (0-1)*

*1 = no lag (snow pack temp=current day air temp) as the lag factor goes to zero, the snow pack's temperature will be less influenced by the current day's air temperature.*

- real \*8 **parm::wt\_shall**  
*shallow water table depth above the impervious layer (mm H2O)*
- real \*8 **parm::sq\_rto**
- real \*8 **parm::qtile**  
*amount of water in drainage tile flow in HRU soil layer for the day (mm H2O)*
- real \*8 **parm::inflpcp**  
*amount of precipitation that infiltrates into soil (enters soil) (mm H2O)*
- real \*8 **parm::fixn**  
*amount of nitrogen added to the plant biomass via fixation on the day in HRU (kg N/ha)*
- real \*8 **parm::latlyr**  
*amount of water in lateral flow in layer in HRU for the day (mm H2O)*
- real \*8 **parm::snofall**  
*amount of precipitation falling as freezing rain/snow on day in HRU (mm H2O)*
- real \*8 **parm::snomlt**  
*amount of water in snow melt for the day in HRU (mm H2O)*
- real \*8 **parm::tloss**  
*amount of water removed from surface runoff via transmission losses on day in HRU (mm H2O)*
- real \*8 **parm::lpndloss**
- real \*8 **parm::lwetloss**
- real \*8 **parm::bioday**  
*biomass generated on current day in HRU (kg)*
- real \*8 **parm::cfertn**  
*total amount of nitrogen applied to soil during continuous fertilizer operation in HRU on day (kg N/ha)*
- real \*8 **parm::cfertp**  
*amount of phosphorus applied to soil during continuous fertilizer operation in HRU on day (kg P/ha)*
- real \*8 **parm::fertn**  
*total amount of nitrogen applied to soil in HRU on day in fertilizer application (kg N/ha)*
- real \*8 **parm::sepcday**  
*micropore percolation from bottom of the soil layer on day in HRU (mm H2O)*
- real \*8 **parm::sol\_rd**  
*current rooting depth (mm)*
- real \*8 **parm::sedrch**  
*sediment transported out of channel or reach during time step (metric tons)*
- real \*8 **parm::sepcrktot**
- real \*8 **parm::fertno3**
- real \*8 **parm::fertnh3**
- real \*8 **parm::fertorgn**
- real \*8 **parm::fertsolp**
- real \*8 **parm::fertorgp**
- real \*8 **parm::qdfr**  
*fraction of water yield that is surface runoff (none)*
- real \*8 **parm::fertp**  
*total amount of phosphorus applied to soil in HRU on day in fertilizer application (kg P/ha)*
- real \*8 **parm::grazn**  
*amount of nitrogen added to soil in grazing on the day in HRU (kg N/ha)*
- real \*8 **parm::grazp**  
*amount of phosphorus added to soil in grazing on the day in HRU (kg P/ha)*
- real \*8 **parm::soxy**  
*saturation dissolved oxygen concentration (mg/L)*

- real \*8 `parm::rtwtr`  
water leaving reach on day ( $m^3 H_2O$ )
- real \*8 `parm::sdti`  
average flow rate in reach for day ( $m^3/s$ )
- real \*8 `parm::ressa`  
surface area of reservoir on day (ha)
- real \*8 `parm::da_km`  
area of the watershed in square kilometers ( $km^2$ )
- real \*8 `parm::rchdep`  
depth of flow on day (m)
- real \*8 `parm::rtevp`  
evaporation from reach on day ( $m^3 H_2O$ )
- real \*8 `parm::rttime`  
reach travel time (hour)
- real \*8 `parm::rttlc`  
transmission losses from reach on day ( $m^3 H_2O$ )
- real \*8 `parm::resflwi`  
water entering reservoir on day ( $m^3 H_2O$ )
- real \*8 `parm::wdprch`  
die-off factor for persistent bacteria in streams (1/day)
- real \*8 `parm::resev`  
evaporation from reservoir on day ( $m^3 H_2O$ )
- real \*8 `parm::resflwo`  
water leaving reservoir on day ( $m^3 H_2O$ )
- real \*8 `parm::respccp`  
precipitation on reservoir for day ( $m^3 H_2O$ )
- real \*8 `parm::ressedi`  
sediment entering reservoir during time step (metric tons)
- real \*8 `parm::ressedo`  
sediment leaving reservoir during time step (metric tons)
- real \*8 `parm::ressep`  
seepage from reservoir on day ( $m^3 H_2O$ )
- real \*8 `parm::pperco_bsn`  
phosphorus percolation coefficient. Ratio of soluble phosphorus in surface to soluble phosphorus in percolate
- real \*8 `parm::nperco_bsn`  
basin nitrate percolation coefficient (0-1)  
0:concentration of nitrate in surface runoff is zero  
1:percolate has same concentration of nitrate as surface runoff
- real \*8 `parm::rsdco`  
residue decomposition coefficient. The fraction of residue which will decompose in a day assuming optimal moisture, temperature, C:N ratio, and C:P ratio
- real \*8 `parm::voltot`  
total volume of cracks expressed as depth per unit area (mm)
- real \*8 `parm::phoskd_bsn`
- real \*8 `parm::msk_x`  
weighting factor controlling relative importance of inflow rate and outflow rate in determining storage on reach
- real \*8 `parm::volcrmin`  
minimum crack volume allowed in any soil layer (mm), or  
minimum soil volume in profile (mm)
- real \*8 `parm::bactkdq`  
bacteria soil partitioning coefficient. Ratio of solution bacteria in surface layer to solution bacteria in runoff soluble and sorbed phase in surface runoff.

- real \*8 `parm::canev`  
*amount of water evaporated from canopy storage (mm H2O)*
- real \*8 `parm::precipday`  
*precipitation, or effective precipitation reaching soil surface, for the current day in HRU (mm H2O)*
- real \*8 `parm::uno3d`  
*plant nitrogen deficiency for day in HRU (kg N/ha)*
- real \*8 `parm::usle`  
*daily soil loss predicted with USLE equation (metric tons/ha)*
- real \*8 `parm::rcn`  
*concentration of nitrogen in the rainfall (mg/L)*
- real \*8 `parm::surlag_bsn`
- real \*8 `parm::thbact`  
*temperature adjustment factor for bacteria die-off/growth*
- real \*8 `parm::wlpq20`  
*overall rate change for less persistent bacteria in soil solution (1/day)*
- real \*8 `parm::wlps20`  
*overall rate change for less persistent bacteria adsorbed to soil particles (1/day)*
- real \*8 `parm::wpq20`  
*overall rate change for persistent bacteria in soil solution (1/day)*
- real \*8 `parm::wps20`  
*overall rate change for persistent bacteria adsorbed to soil particles (1/day)*
- real \*8 `parm::bactrop`  
*persistent bacteria transported to main channel with surface runoff (# colonies/ha)*
- real \*8 `parm::bactsedp`  
*persistent bacteria transported with sediment in surface runoff (# colonies/ha)*
- real \*8 `parm::enratio`  
*enrichment ratio calculated for current day in HRU (none)*
- real \*8 `parm::pndpcp`  
*precipitation on pond during day ( $m^3$  H2O)*
- real \*8 `parm::wetpcp`  
*precipitation on wetland for day ( $m^3$  H2O)*
- real \*8 `parm::wetsep`  
*seepage from wetland bottom for day ( $m^3$  H2O)*
- real \*8 `parm::pndev`  
*evaporation from pond on day ( $m^3$  H2O)*
- real \*8 `parm::pndflwi`  
*volume of water flowing into pond on day ( $m^3$  H2O)*
- real \*8 `parm::pndsedo`  
*sediment leaving pond during day (metric tons)*
- real \*8 `parm::pndsep`  
*seepage from pond on day ( $m^3$  H2O)*
- real \*8 `parm::wetev`  
*evaporation from wetland for day ( $m^3$  H2O)*
- real \*8 `parm::wetflwi`  
*volume of water flowing in wetland on day ( $m^3$  H2O)*
- real \*8 `parm::wetsedo`  
*sediment loading from wetland for day (metric tons)*
- real \*8 `parm::da_ha`  
*drainage area of watershed in hectares (ha)*
- real \*8 `parm::pndflwo`  
*volume of water flowing out of pond on day ( $m^3$  H2O)*

- real \*8 `parm::vpd`  
vapor pressure deficit (kPa)
- real \*8 `parm::wetflwo`  
volume of water flowing out wetland on day ( $m^3 H_2O$ )
- real \*8 `parm::wetsedi`  
sediment loading to wetland for day (metric tons)
- real \*8 `parm::evlai`  
leaf area index at which no evaporation occurs. This variable is used in ponded HRUs (eg rice) where evaporation from the water surface is restricted by the plant canopy cover. Evaporation from the water surface equals potential ET when LAI = 0 and decreased linearly to 0 when LAI = EVLAI
- real \*8 `parm::evrch`  
Reach evaporation adjustment factor. Evaporation from the reach is multiplied by EVRCH. This variable was created to limit the evaporation predicted in arid regions.
- real \*8 `parm::ep_day`  
actual amount of transpiration that occurs on day in HRU (mm H<sub>2</sub>O)
- real \*8 `parm::pet_day`  
potential evapotranspiration on current day in HRU (mm H<sub>2</sub>O)
- real \*8 `parm::bactrolp`  
less persistent bacteria transported to main channel with surface runoff (# colonies/ha)
- real \*8 `parm::bactsedlp`  
less persistent bacteria transported with sediment in surface runoff (# colonies/ha)
- real \*8 `parm::adj_pkr`  
peak rate adjustment factor in the subbasin. Used in the MUSLE equation to account for impact of peak flow on erosion (none)
- real \*8 `parm::n_updis`  
nitrogen uptake distribution parameter. This parameter controls the amount of nitrogen removed from the different soil layer layers by the plant. In particular, this parameter allows the amount of nitrogen removed from the surface layer via plant uptake to be controlled. While the relationship between UBN and N removed from the surface layer is affected by the depth of the soil profile, in general, as UBN increases the amount of N removed from the surface layer relative to the amount removed from the entire profile increases
- real \*8 `parm::nactfr`  
nitrogen active pool fraction. The fraction of organic nitrogen in the active pool (none)
- real \*8 `parm::p_updis`  
phosphorus uptake distribution parameter This parameter controls the amount of phosphorus removed from the different soil layers by the plant. In particular, this parameter allows the amount of phosphorus removed from the surface layer via plant uptake to be controlled. While the relationship between UBP and P uptake from the surface layer is affected by the depth of the soil profile, in general, as UBP increases the amount of P removed from the surface layer relative to the amount removed from the entire profile increases
- real \*8 `parm::snoev`  
amount of water in snow lost through sublimation on current day in HRU (mm H<sub>2</sub>O)
- real \*8 `parm::sno3up`  
amount of nitrate moving upward in the soil profile in watershed (kg N/ha)
- real \*8 `parm::reactw`  
amount of pesticide in lake water of reach that is lost through reactions (mg pst)
- real \*8 `parm::es_day`  
actual amount of evaporation (soil et) that occurs on day in HRU (mm H<sub>2</sub>O)
- real \*8 `parm::sdiagro1pq`  
average annual change in the number of less persistent bacteria colonies in soil solution in watershed (# cfu/m<sup>2</sup>)
- real \*8 `parm::sdiagro1ps`  
average annual change in the number of less persistent bacteria colonies on soil particles in watershed (# cfu/m<sup>2</sup>)
- real \*8 `parm::sdiagro2pq`  
average annual change in the number of persistent bacteria colonies in soil solution in watershed (# cfu/m<sup>2</sup>)
- real \*8 `parm::sdiagro2ps`

- average annual change in the number of persistent bacteria colonies on soil particles in watershed (# cfu/m<sup>2</sup>)*

  - real \*8 **parm::wof\_lp**

*fraction for less persistent bacteria on foliage that is washed off by a rainfall event (none)*

  - real \*8 **parm::ep\_max**

*maximum amount of transpiration (plant et) that can occur on day in HRU (mm H2O)*

  - real \*8 **parm::sbactrolp**

*average annual number of less persistent bacteria transported to main channel with surface runoff in solution (# colonies/ha)*

  - real \*8 **parm::sbactrop**

*average annual number of persistent bacteria transported to main channel with surface runoff in solution (# colonies/ha)*

  - real \*8 **parm::sbactsedlp**

*average annual number of less persistent bacteria transported with sediment in surface runoff (# colonies/ha)*

  - real \*8 **parm::sbactsedp**

*average annual number of persistent bacteria transported with sediment in surface runoff (# colonies/ha)*

  - real \*8 **parm::sbactlchlp**

*average annual number of less persistent bacteria lost from soil surface layer by percolation (# cfu/m<sup>2</sup>)*

  - real \*8 **parm::sbactlchp**

*average annual number of persistent bacteria lost from soil surface layer by percolation (# cfu/m<sup>2</sup>)*

  - real \*8 **parm::rchwtr**

*water stored in reach at beginning of day (m<sup>3</sup> H2O)*

  - real \*8 **parm::resuspst**

*amount of pesticide moving from sediment to reach due to resuspension (mg pst)*

  - real \*8 **parm::setlpst**

*amount of pesticide moving from water to sediment due to settling (mg pst)*

  - real \*8 **parm::psp\_bsn**
  - real \*8 **parm::bsprev**

*surface runoff lagged from prior day of simulation (mm H2O)*

  - real \*8 **parm::bssprev**

*lateral flow lagged from prior day of simulation (mm H2O)*

  - real \*8 **parm::spadyev**

*average annual amount of water removed from potholes by evaporation in watershed (mm H2O)*

  - real \*8 **parm::spadyo**

*average annual amount of water released to main channel from potholes in watershed (mm H2O)*

  - real \*8 **parm::spadyrfv**

*average annual amount of precipitation on potholes in watershed (mm H2O)*

  - real \*8 **parm::spadysp**

*average annual amount of water removed from potholes by seepage in watershed (mm H2O)*

  - real \*8 **parm::spadyosp**
  - real \*8 **parm::qday**

*amount of surface runoff loading to main channel from HRU on current day (includes effects of transmission losses) (mm H2O)*

  - real \*8 **parm::al5**

*fraction of total rainfall that occurs during 0.5h of highest intensity rain (none)*

  - real \*8 **parm::no3pcp**

*nitrate added to the soil in rainfall (kg N/ha)*

  - real \*8 **parm::pndsedc**

*net change in sediment in pond during day (metric tons)*

  - real \*8 **parm::usle\_ei**

*USLE rainfall erosion index on day for HRU (100(ft-tn in)/(acre-hr))*

  - real \*8 **parm::rcharea**

- cross-sectional area of flow ( $m^2$ )*

  - real \*8 `parm::volatpst`  
*amount of pesticide lost from lake water of reach by volatilization (mg pst)*
- real \*8 `parm::ubw`  
*water uptake distribution parameter. This parameter controls the amount of water removed from the different soil layers by the plant. In particular, this parameter allows the amount of water removed from the surface layer via plant uptake to be controlled. While the relationship between UBW and H2O removed from the surface layer is affected by the depth of the soil profile, in general, as UBW increases the amount of water removed from the surface layer relative to the amount removed from the entire profile increases*
- real \*8 `parm::uobn`  
*nitrogen uptake normalization parameter. This variable normalizes the nitrogen uptake so that the model can easily verify that uptake from the different soil layers sums to 1.0*
- real \*8 `parm::uobp`  
*phosphorus uptake normalization parameter. This variable normalizes the phosphorus uptake so that the model can easily verify that uptake from the different soil layers sums to 1.0*
- real \*8 `parm::uobw`  
*water uptake normalization parameter. This variable normalizes the water uptake so that the model can easily verify that uptake from the different soil layers sums to 1.0*
- real \*8 `parm::wetsedc`  
*net change in sediment in wetland during day (metric tons)*
- real \*8 `parm::respesti`  
*pesticide entering reservoir on day (mg pst)*
- real \*8 `parm::rcor`  
*correction coefficient for generated rainfall to ensure that the annual means for generated and observed values are comparable (needed only if IDIST=1)*
- real \*8 `parm::rexp`  
*value of exponent for mixed exponential rainfall distribution (needed only if IDIST=1)*
- real \*8 `parm::snocov1`  
*1st shape parameter for snow cover equation. This parameter is determined by solving the equation for 50% snow cover*
- real \*8 `parm::snocov2`  
*2nd shape parameter for snow cover equation. This parameter is determined by solving the equation for 95% snow cover*
- real \*8 `parm::snocovmx`  
*Minimum snow water content that corresponds to 100% snow cover. If the snow water content is less than SNOC $\leftrightarrow$  OVMX, then a certain percentage of the ground will be bare (mm H2O)*
- real \*8 `parm::ai0`  
*ratio of chlorophyll-a to algal biomass (ug chla/mg alg)*
- real \*8 `parm::ai1`  
*fraction of algal biomass that is nitrogen (mg N/mg alg)*
- real \*8 `parm::ai2`  
*fraction of algal biomass that is phosphorus (mg P/mg alg)*
- real \*8 `parm::ai3`  
*the rate of oxygen production per unit of algal photosynthesis (mg O2/mg alg)*
- real \*8 `parm::ai4`  
*the rate of oxygen uptake per unit of algae respiration (mg O2/mg alg)*
- real \*8 `parm::ai5`  
*the rate of oxygen uptake per unit of NH3 nitrogen oxidation (mg O2/mg N)*
- real \*8 `parm::ai6`  
*the rate of oxygen uptake per unit of NO2 nitrogen oxidation (mg O2/mg N)*
- real \*8 `parm::rhoq`  
*algal respiration rate at 20 deg C (1/day or 1/hr)*
- real \*8 `parm::tfact`

- fraction of solar radiation computed in the temperature heat balance that is photosynthetically active*
- real \*8 `parm::k_l`  
*half-saturation coefficient for light (MJ/(m<sup>2</sup>\*hr))*
- real \*8 `parm::k_n`  
*michaelis-menton half-saturation constant for nitrogen (mg N/L)*
- real \*8 `parm::k_p`  
*michaelis-menton half saturation constant for phosphorus (mg P/L)*
- real \*8 `parm::lambda0`  
*non-algal portion of the light extinction coefficient (1/m)*
- real \*8 `parm::lambda1`  
*linear algal self-shading coefficient (1/(m\*ug chla/L))*
- real \*8 `parm::lambda2`  
*nonlinear algal self-shading coefficient ((1/m)(ug chla/L)\*\*(-2/3))*
- real \*8 `parm::mumax`  
*maximum specific algal growth rate at 20 deg C(1/day or 1/hr)*
- real \*8 `parm::p_n`  
*algal preference factor for ammonia*
- real \*8 `parm::rnum1`  
*variable to hold value for rnum1s(:) (none)*
- real \*8 `parm::etday`  
*actual evapotranspiration occurring on day in HRU (mm H<sub>2</sub>O)*
- real \*8 `parm::auton`  
*amount of nitrogen applied in auto-fert application (kg N/ha)*
- real \*8 `parm::autop`  
*amount of phosphorus applied in auto-fert application (kg P/ha)*
- real \*8 `parm::hmntl`  
*amount of nitrogen moving from active organic to nitrate pool in soil profile on current day in HRU (kg N/ha)*
- real \*8 `parm::hmptl`  
*amount of phosphorus moving from active organic to nitrate pool in soil profile on current day in HRU (kg P/ha)*
- real \*8 `parm::rmn2tl`  
*amount of nitrogen moving from the fresh organic (residue) to the nitrate(80%) and active organic(20%) pools in soil profile on current day in HRU (kg N/ha)*
- real \*8 `parm::rwntl`  
*amount of nitrogen moving from active organic to stable organic pool in soil profile on current day in HRU (kg N/ha)*
- real \*8 `parm::gwseep`  
*amount of water recharging deep aquifer on current day in HRU (mm H<sub>2</sub>O)*
- real \*8 `parm::revapday`  
*amount of water moving from the shallow aquifer into the soil profile or being taken up by plant roots in the shallow aquifer or in the bank storage zone (mm H<sub>2</sub>O)*
- real \*8 `parm::rmp1tl`  
*amount of phosphorus moving from the labile mineral pool to the active mineral pool in the soil profile on the current day in the HRU (kg P/ha)*
- real \*8 `parm::rmptl`  
*amount of phosphorus moving from the fresh organic (residue) to the labile(80%) and organic(20%) pools in soil profile on current day in HRU (kg P/ha)*
- real \*8 `parm::roctl`  
*amount of phosphorus moving from the active mineral pool to the stable mineral pool in the soil profile on the current day in the HRU (kg P/ha)*
- real \*8 `parm::wdntl`  
*amount of nitrogen lost from nitrate pool by denitrification in soil profile on current day in HRU (kg N/ha)*
- real \*8 `parm::cmn_bsn`
- real \*8 `parm::wdlprch`

- die-off factor for less persistent bacteria in streams (1/day)*
- real \*8 [parm::wdpres](#)
  - die-off factor for persistent bacteria in reservoirs (1/day)*
- real \*8 [parm::petmeas](#)
  - potential ET value read in for day (mm H2O)*
- real \*8 [parm::bury](#)
  - loss of pesticide from active sediment layer by burial (mg pst)*
- real \*8 [parm::difus](#)
  - diffusion of pesticide from sediment to reach lake water (mg pst)*
- real \*8 [parm::reactb](#)
  - amount of pesticide in sediment that is lost through reactions (mg pst)*
- real \*8 [parm::solpesto](#)
  - soluble pesticide concentration in outflow on day (mg pst/m<sup>3</sup>)*
- real \*8 [parm::wdlpres](#)
  - die-off factor for less persistent bacteria in reservoirs (1/day)*
- real \*8 [parm::sorpesto](#)
  - sorbed pesticide concentration in outflow on day (mg pst/m<sup>3</sup>)*
- real \*8 [parm::solpesti](#)
  - soluble pesticide entering reservoir (mg pst)*
- real \*8 [parm::sorpesti](#)
  - sorbed pesticide entering reservoir (mg pst)*
- real \*8 [parm::spcon\\_bsn](#)
- real \*8 [parm::spexp\\_bsn](#)
- real \*8 [parm::msk\\_co1](#)
  - calibration coefficient to control impact of the storage time constant for the reach at bankfull depth (phi(10,:)) upon the storage time constant for the reach used in the Muskingum flow method*
- real \*8 [parm::msk\\_co2](#)
  - calibration coefficient to control impact of the storage time constant for the reach at 0.1 bankfull depth (phi(13,:)) upon the storage time constant for the reach used in the Muskingum flow method*
- real \*8 [parm::deepstp](#)
  - depth of water in deep aquifer in HRU (mm H2O)*
- real \*8 [parm::shallstp](#)
  - depth of water in shallow aquifer in HRU on previous day (mm H2O)*
- real \*8 [parm::snoprev](#)
  - amount of water stored as snow on previous day (mm H2O)*
- real \*8 [parm::swprev](#)
  - amount of water stored in soil profile in the HRU on the previous day (mm H2O)*
- real \*8 [parm::reschlao](#)
  - amount of chlorophyll-a leaving reservoir on day (kg chl-a)*
- real \*8 [parm::resno2o](#)
  - amount of nitrite leaving reservoir on day (kg N)*
- real \*8 [parm::resno3o](#)
  - amount of nitrate leaving reservoir on day (kg N)*
- real \*8 [parm::resorgno](#)
  - amount of organic N leaving reservoir on day (kg N)*
- real \*8 [parm::resorgpo](#)
  - amount of organic P leaving reservoir on day (kg P)*
- real \*8 [parm::ressolpo](#)
  - amount of soluble P leaving reservoir on day (kg P)*
- real \*8 [parm::resnh3o](#)
  - amount of ammonia leaving reservoir on day (kg N)*



- real \*8 **parm::bactminlp**  
*Threshold detection level for less persistent bacteria. When bacteria levels drop to this amount the model considers bacteria in the soil to be insignificant and sets the levels to zero (cfu/m<sup>2</sup>)*
- real \*8 **parm::bactminp**  
*Threshold detection level for persistent bacteria. When bacteria levels drop to this amount the model considers bacteria in the soil to be insignificant and sets the levels to zero (cfu/m<sup>2</sup>)*
- real \*8 **parm::trnsrch**  
*fraction of transmission losses from main channel that enter deep aquifer*
- real \*8 **parm::wp20p\_plt**  
*overall rate change for persistent bacteria on foliage (1/day)*
- real \*8 **parm::potsedo**  
*sediment leaving pothole to main channel from HRU on day (metric tons/ha)*
- real \*8 **parm::pest\_sol**
- real \*8 **parm::bact\_swf**  
*fraction of manure containing active colony forming units (cfu)*
- real \*8 **parm::bactmx**  
*bacteria percolation coefficient. Ratio of solution bacteria in surface layer to solution bacteria in percolate*
- real \*8 **parm::cncoef**  
*plant ET curve number coefficient*
- real \*8 **parm::wp20lp\_plt**  
*overall rate change for less persistent bacteria on foliage (1/day)*
- real \*8 **parm::cdn\_bsn**
- real \*8 **parm::sdnco\_bsn**
- real \*8 **parm::cn\_froz**  
*drainage coefficient (mm day<sup>-1</sup>)*
- real \*8 **parm::dorm\_hr**  
*time threshold used to define dormant (hours)*
- real \*8 **parm::smxco**  
*adjustment factor for max curve number s factor (0-1)*
- real \*8 **parm::tb\_adj**  
*adjustment factor for subdaily unit hydrograph basetime*
- real \*8 **parm::chla\_subco**  
*regional adjustment on sub chla\_a loading (fraction)*
- real \*8 **parm::depimp\_bsn**  
*depth to impervious layer. Used to model perched water tables in all HRUs in watershed (mm)*
- real \*8 **parm::ddrain\_bsn**  
*depth to the sub-surface drain (mm)*
- real \*8 **parm::rch\_san**
- real \*8 **parm::rch\_sil**
- real \*8 **parm::rch\_cla**
- real \*8 **parm::rch\_sag**
- real \*8 **parm::rch\_lag**
- real \*8 **parm::rch\_gra**
- real \*8 **parm::hlife\_ngw\_bsn**  
*Half-life of nitrogen in groundwater? (days)*
- real \*8 **parm::ch\_opco\_bsn**
- real \*8 **parm::ch\_onco\_bsn**
- real \*8 **parm::decr\_min**  
*Minimum daily residue decay.*
- real \*8 **parm::rcn\_sub\_bsn**  
*Concentration of nitrogen in the rainfall (mg/kg)*
- real \*8 **parm::bc1\_bsn**

- real \*8 **parm::bc2\_bsn**
- real \*8 **parm::bc3\_bsn**
- real \*8 **parm::bc4\_bsn**
- real \*8 **parm::anion\_excl\_bsn**
- real \*8, dimension(:), allocatable **parm::wat\_tbl**  
*water table based on depth from soil surface (mm)*
- real \*8, dimension(:,:), allocatable **parm::vwt**
- real \*8 **parm::re\_bsn**  
*Effective radius of drains (range 3.0 - 40.0) (mm)*
- real \*8 **parm::sdrain\_bsn**  
*Distance bewtween two drain or tile tubes (range 7600.0 - 30000.0) (mm)*
- real \*8 **parm::sstmaxd\_bsn**
- real \*8 **parm::drain\_co\_bsn**  
*Drainage coeffcient (range 10.0 - 51.0) (mm-day-1)*
- real \*8 **parm::latksatf\_bsn**  
*Multiplication factor to determine lateral ksat from SWAT ksat input value for HRU (range 0.01 - 4.0)*
- real \*8 **parm::pc\_bsn**  
*Pump capacity (def val = 1.042 mm h-1 or 25 mm day-1) (mm h-1)*
- integer **parm::i\_subhw**
- integer **parm::imgt**
- integer **parm::iwtr**
- integer **parm::mo\_atmo**
- integer **parm::mo\_atmo1**
- integer **parm::iyr\_atmo1**
- integer **parm::matmo**
- integer **parm::mch**  
*maximum number of channels*
- integer **parm::mcr**  
*maximum number of crops grown per year*
- integer **parm::mcldb**  
*maximum number of crops/landcover in database file (crop.dat)*
- integer **parm::mfdb**  
*maximum number of fertilizers in fert.dat*
- integer **parm::mhru**  
*maximum number of HRUs in watershed*
- integer **parm::mhyd**  
*maximum number of hydrograph nodes*
- integer **parm::mpdb**  
*maximum number of pesticides in pest.dat*
- integer **parm::mrg**  
*maximum number of rainfall/temp gages (none)*
- integer **parm::mgr**  
*maximum number of grazings per year*
- integer **parm::mnr**  
*maximum number of years of rotation*
- integer **parm::myr**  
*maximum number of years of simulation*
- integer **parm::isubwq**  
*subbasin water quality code*  
*0 do not calculate algae/CBOD 1 calculate algae/CBOD drainmod tile equations*
- integer **parm::ffcst**
- integer **parm::isproj**

- special project code (none):*
  - 1 test rewind (run simulation twice)*
- integer **parm::nbyr**
  - number of calendar years simulated (none)*
- integer **parm::irte**
  - water routing method (none):*
  - 0 variable storage method*
  - 1 Muskingum method*
- integer **parm::nrch**
  - number of reaches in watershed (none)*
- integer **parm::nres**
  - total number of reservoirs in watershed (none)*
- integer **parm::nhru**
  - number of last HRU in previous subbasin or*
  - number of HRUs in watershed (none)*
- integer **parm::i\_mo**
  - current month being simulated or month of next day of simulation (none)*
- integer **parm::immo**
  - current cumulative month of simulation (none)*
- integer **parm::wndsim**
  - wind speed input code (noen)*
  - 1 measured data read for each subbasin*
  - 2 data simulated for each subbasin*
- integer **parm::ihout**
  - variable to hold value for ihouts(:) (none)*
- integer **parm::inum2**
  - variable to hold value for inum2s(:) (none)*
- integer **parm::inum3**
  - variable to hold value for inum3s(:) (none)*
- integer **parm::inum4**
  - variable to hold value for inum4s(:) (none)*
- integer **parm::icfac**
  - icfac = 0 for C-factor calculation using Cmin (as described in manual)*
  - = 1 for new C-factor calculation from RUSLE (no minimum needed)*
- integer **parm::inum5**
- integer **parm::inum6**
- integer **parm::inum7**
- integer **parm::inum8**
- integer **parm::mrech**
  - maximum number of rechour files*
- integer **parm::nrgage**
  - number of raingage files (none)*
- integer **parm::nrgfil**
  - number of rain gages per file (none)*
- integer **parm::nrtot**
  - total number of rain gages (none)*
- integer **parm::ntgage**
  - number of temperature gage files (none)*
- integer **parm::ntgfil**
  - number of temperature gages per file (none)*
- integer **parm::nttot**
  - total number of temperature gages (none)*

- integer `parm::tmpsim`  
*temperature input code (none)*  
*1 measured data read for each subbasin*  
*2 data simulated for each subbasin*
- integer `parm::icrk`  
*crack flow code*  
*1: simulate crack flow in watershed*
- integer `parm::irtpest`  
*number of pesticide to be routed through the watershed. Redefined to the sequence number of pesticide in NPNO(:) which is to be routed through the watershed (none)*
- integer `parm::igropt`  
*Qual2E option for calculating the local specific growth rate of algae*  
*1: multiplicative.*
- integer `parm::npmx`  
*number of different pesticides used in the simulation (none)*
- integer `parm::curyr`  
*current year in simulation (sequence) (none)*
- integer `parm::itdrn`  
*tile drainage equations flag/code*  
*1 simulate tile flow using subroutine drains(wt\_shall)*  
*0 simulate tile flow using subroutine origtile(wt\_shall,d)*
- integer `parm::iwt dn`  
*water table depth algorithms flag/code*  
*1 simulate wt\_shall using subroutine new water table depth routine*  
*0 simulate wt\_shall using subroutine original water table depth routine*
- integer `parm::ismax`  
*maximum depressional storage selection flag/code (none)*  
*0 = static depressional storage (stmaxd) read from .bsn for the global value or .sdr for specific HRUs*  
*1 = dynamic storage (stmaxd) based on random roughness, tillage and cumulative rainfall intensity by [depstor.f90](#)*
- integer `parm::iroutunit`  
*not being implemented in this version drainmod tile equations*
- integer `parm::ires_nut`
- integer `parm::iclb`  
*auto-calibration flag*
- integer `parm::mrecc`  
*maximum number of reccnst files*
- integer `parm::mrecd`  
*maximum number of recday files*
- integer `parm::mrecm`  
*maximum number of recmon files*
- integer `parm::mtill`  
*max number of tillage types in till.dat*
- integer `parm::mudb`  
*maximum number of urban land types in urban.dat*
- integer `parm::idist`  
*rainfall distribution code*  
*0 for skewed normal dist*  
*1 for mixed exponential distribution*
- integer `parm::mrecy`  
*maximum number of recyear files*
- integer `parm::nyskip`  
*number of years to skip output summarization and printing (none)*
- integer `parm::slrsim`

- solar radiation input code (none)*
  - 1 measured data read for each subbasin*
  - 2 data simulated for each subbasin*
- integer **parm::ideg**
  - channel degradation code*
  - 0: do not compute channel degradation*
  - 1: compute channel degradation (downcutting and widening)*
- integer **parm::ievent**
  - rainfall/runoff code (none)*
  - 0 daily rainfall/curve number technique 1 daily rainfall/curve number technique/ daily routing 2 sub-daily rainfall /↔ Green&Ampt technique/ daily routing 3 sub-daily rainfall /Green&Ampt technique/ hourly routing*
- integer **parm::ipet**
  - code for potential ET method (none)*
  - 0 Priestley-Taylor method*
  - 1 Penman/Monteith method*
  - 2 Hargreaves method*
  - 3 read in daily potential ET data*
- integer **parm::iopera**
- integer **parm::idaf**
  - beginning day of simulation (julian date)*
- integer **parm::idal**
  - ending day of simulation (julian date)*
- integer **parm::rhsim**
  - relative humidity input code (none)*
  - 1 measured data read for each subbasin*
  - 2 data simulated for each subbasin*
- integer **parm::leapyr**
  - leap year flag (none)*
  - 0 leap year*
  - 1 regular year*
- integer **parm::id1**
  - first day of simulation in current year (julian date)*
- integer **parm::mo\_chk**
  - check for month being simulated; when mo\_chk differs from mo, monthly output is printed (none)*
- integer **parm::nhtot**
  - total number of relative humidity records in file*
- integer **parm::nstot**
  - total number of solar radiation records in file (none)*
- integer **parm::nwtot**
  - total number of wind speed records in file*
- integer **parm::ifirsts**
  - solar radiation data search code (none)*
  - 0 first day of solar radiation data located in file*
  - 1 first day of solar radiation data not located in file*
- integer **parm::ifirsth**
  - relative humidity data search code (none)*
  - 0 first day of relative humidity data located in file*
  - 1 first day of relative humidity data not located in file*
- integer **parm::ifirstw**
  - wind speed data search code (none)*
  - 0 first day of wind speed data located in file*
  - 1 first day of wind speed data not located in file*
- integer **parm::ilog**
  - streamflow print code (none)*
  - 0 print streamflow in reach*
  - 1 print Log10 streamflow in reach*

- integer **parm::itotr**  
*number of output variables printed (output.rch)*
- integer **parm::iyr**  
*current year of simulation (year)*
- integer **parm::iwq**  
*stream water quality code*  
*0 do not model stream water quality*  
*1 model stream water quality (QUAL2E & pesticide transformations)*
- integer **parm::iskip**  
*flag for calculations performed only for the first year of simulation (none)*
- integer **parm::ifirstpet**  
*potential ET data search code (none)*  
*0 first day of potential ET data located in file*  
*1 first day of potential ET data not located in file*
- integer **parm::iprp**  
*print code for output.pst file*  
*0 do not print pesticide output*  
*1 print pesticide output*
- integer **parm::itotb**  
*number of output variables printed (output.sub)*
- integer **parm::itots**  
*number of output variables printed (output.hru)*
- integer **parm::itoth**  
*number of HRUs printed (output.hru/output.wtr)*
- integer **parm::pcpsim**  
*rainfall input code (none)*  
*1 measured data read for each subbasin*  
*2 data simulated for each subbasin*
- integer **parm::nd\_30**
- integer **parm::iphr**
- integer **parm::isto**
- integer **parm::isol**
- integer **parm::fcstcycles**  
*number of times forecast period is simulated (using different weather generator seeds each time)*
- integer **parm::fcstday**  
*beginning date of forecast period (julian date)*
- integer **parm::fcstyr**  
*beginning year of forecast period*
- integer **parm::iscen**  
*scenarios counter*
- integer **parm::subtot**  
*number of subbasins in watershed (none)*
- integer **parm::ogen**  
*random number generator seed code (none)*
- integer **parm::mlyr**  
*maximum number of soil layers*
- integer **parm::mpst**  
*max number of pesticides used in wshed*
- integer **parm::mres**  
*maximum number of reservoirs*
- integer **parm::msub**  
*maximum number of subbasins*
- integer **parm::igen**

- random number generator seed code (none):*
  - 0: use default numbers*
  - 1: generate new numbers in every simulation*
- integer **parm::iprint**
  - print code (none): 0=monthly, 1=daily, 2=annually*
- integer **parm::iida**
  - day being simulated (current julian date) (julian date)*
- integer **parm::icn**
  - CN method flag (for testing alternative method):*
  - 0 use traditional SWAT method which bases CN on soil moisture*
  - 1 use alternative method which bases CN on plant ET*
  - 2 use traditional SWAT method which bases CN on soil moisture but rention is adjusted for mildly-sloped tiled-drained watersheds.*
- integer **parm::ised\_det**
  - max half-hour rainfall fraction calc option:*
  - 0 generate max half-hour rainfall fraction from triangular distribution*
  - 1 use monthly mean max half-hour rainfall fraction*
- integer **parm::fcstcnt**
- integer **parm::idtil**
- integer, dimension(100) **parm::ida\_lup**
- integer, dimension(100) **parm::iyr\_lup**
- integer **parm::no\_lup**
- integer **parm::nostep**
- character(len=13) **parm::rhfile**
  - relative humidity file name (.hmd)*
- character(len=13) **parm::slrfile**
  - solar radiation file name (.slr)*
- character(len=13) **parm::wndfile**
  - wind speed file name (.wnd)*
- character(len=13) **parm::petfile**
  - potential ET file name (.pet)*
- character(len=13) **parm::atmofile**
- character(len=13) **parm::septdb**
  - name of septic tank database file (septwq1.dat)*
- integer, dimension(9) **parm::idg**
  - array location of random number seed used for a given process*
- integer, dimension(:), allocatable **parm::ifirsthr**
  - measured data search code (none)*
  - 0 first day of measured data located in file*
  - 1 first day of measured data not located in file*
- integer, dimension(:), allocatable **parm::ifirstr**
  - measured data search code (none)*
  - 0 first day of measured data located in file*
  - 1 first day of measured data not located in file*
- integer, dimension(8) **parm::values**
  - values(1): year simulation is performed*
  - values(2): month simulation is performed*
  - values(3): day in month simulation is performed*
  - values(4): time difference with respect to Coordinated Universal Time (ie Greenwich Mean Time)*
  - values(5): hour simulation is performed*
  - values(6): minute simulation is performed*
  - values(7): second simulation is performed*
  - values(8): millisecond simulation is performed*
- integer, dimension(13) **parm::ndays**

*julian date for last day of preceding month (where the array location is the number of the month). The dates are for leap years (julian date)*

- integer **parm::mapex**
- real \*8, dimension(:), allocatable [parm::hi\\_targ](#)  
*harvest index target of cover defined at planting ((kg/ha)/(kg/ha))*
- real \*8, dimension(:), allocatable [parm::bio\\_targ](#)  
*biomass target (kg/ha)*
- real \*8, dimension(:), allocatable [parm::tnyld](#)  
*modifier for autofertilization target nitrogen content for plant (kg N/kg yield)*
- integer, dimension(:), allocatable **parm::ifirsta**
- integer, dimension(100) **parm::mo\_transb**
- integer, dimension(100) **parm::mo\_transe**
- integer, dimension(100) **parm::ih\_tran**
- integer [parm::msdb](#)  
*maximum number of sept wq data database (none)*
- integer **parm::iseptic**
- real \*8, dimension(:), allocatable [parm::sptqs](#)  
*flow rate of the septic tank effluent per capita (m3/d)*
- real \*8, dimension(:), allocatable [parm::sptbodconcs](#)  
*Biological Oxygen Demand of the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::spttssconcs](#)  
*concentration of total suspended solid in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptnh4concs](#)  
*concentration of total phosphorus of the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptno3concs](#)  
*concentration of nitrate in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptno2concs](#)  
*concentration of nitrite in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptorgnconcs](#)  
*concentration of organic nitrogen in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptminps](#)  
*concentration of mineral phosphorus in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptorgps](#)  
*concentration of organic phosphorus in the septic tank effluent (mg/l)*
- real \*8, dimension(:), allocatable [parm::sptfcolis](#)  
*concentration of the fecal caliform in the septic tank effluent (cfu/100ml)*
- real \*8, dimension(:), allocatable **parm::failyr**
- real \*8, dimension(:), allocatable **parm::qstemm**
- real \*8, dimension(:), allocatable [parm::bio\\_bod](#)  
*BOD concentration in biozone (kg/ha)*
- real \*8, dimension(:), allocatable [parm::biom](#)  
*biomass of live bacteria in biozone (kg/ha)*
- real \*8, dimension(:), allocatable [parm::rbiom](#)  
*daily change in biomass of live bacteria (kg/ha)*
- real \*8, dimension(:), allocatable [parm::fcoli](#)  
*concentration of the fecal coliform in the biozone septic tank effluent (cfu/100ml)*
- real \*8, dimension(:), allocatable **parm::bz\_perc**
- real \*8, dimension(:), allocatable [parm::plqm](#)  
*plaque in biozone (kg/ha)*
- real \*8, dimension(:), allocatable **parm::bz\_area**
- real \*8, dimension(:), allocatable [parm::bz\\_z](#)  
*depth of biozone layer (mm)*



- real \*8, dimension(:), allocatable [parm::bz\\_thk](#)  
*thickness of biozone (mm)*
- real \*8, dimension(:), allocatable [parm::bio\\_bd](#)  
*density of biomass (kg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable [parm::cmup\\_kgh](#)  
*current soil carbon for first soil layer (kg/ha)*
- real \*8, dimension(:), allocatable [parm::cmtot\\_kgh](#)  
*current soil carbon integrated - aggregating (kg/ha)*
- real \*8, dimension(:), allocatable [parm::coeff\\_denitr](#)  
*denitrification rate coefficient (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_bod\\_dc](#)  
*BOD decay rate coefficient (m<sup>3</sup>/day)*
- real \*8, dimension(:), allocatable [parm::coeff\\_bod\\_conv](#)  
*BOD to live bacteria biomass conversion factor (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_fc1](#)  
*field capacity calibration parameter 1 (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_fc2](#)  
*field capacity calibration parameter 2 (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_fecal](#)  
*fecal coliform bacteria decay rate coefficient (m<sup>3</sup>/day)*
- real \*8, dimension(:), allocatable [parm::coeff\\_mrt](#)  
*mortality rate coefficient (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_nitr](#)  
*nitrification rate coefficient (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_plq](#)  
*conversion factor for plaque from TDS (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_rsp](#)  
*respiration rate coefficient (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_slg1](#)  
*slough-off calibration parameter (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_slg2](#)  
*slough-off calibration parameter (none)*
- real \*8, dimension(:), allocatable [parm::coeff\\_pdistrib](#)
- real \*8, dimension(:), allocatable [parm::coeff\\_solpslp](#)
- real \*8, dimension(:), allocatable [parm::coeff\\_solpintc](#)
- real \*8, dimension(:), allocatable [parm::coeff\\_psorpmx](#)
- integer, dimension(:), allocatable [parm::isep\\_typ](#)  
*septic system type (none)*
- integer, dimension(:), allocatable [parm::i\\_sep](#)  
*soil layer where biozone exists (none)*
- integer, dimension(:), allocatable [parm::isep\\_opt](#)  
*septic system operation flag (1=active, 2=failing, 3 or 0=not operated) (none)*
- integer, dimension(:), allocatable [parm::sep\\_tsincefail](#)
- integer, dimension(:), allocatable [parm::isep\\_tfail](#)
- integer, dimension(:), allocatable [parm::isep\\_iyr](#)
- real \*8, dimension(:), allocatable [parm::sol\\_sumno3](#)
- real \*8, dimension(:), allocatable [parm::sol\\_sumsolp](#)
- real \*8, dimension(:), allocatable [parm::strsw\\_sum](#)
- real \*8, dimension(:), allocatable [parm::strstmp\\_sum](#)
- real \*8, dimension(:), allocatable [parm::strsn\\_sum](#)
- real \*8, dimension(:), allocatable [parm::strsp\\_sum](#)
- real \*8, dimension(:), allocatable [parm::strsa\\_sum](#)

- real \*8, dimension(:), allocatable **parm::pot\_seep**
- real \*8, dimension(:), allocatable **parm::pot\_solp**  
*soluble P loss rate in the pothole (.01 - 0.5) (1/d)*
- real \*8, dimension(:), allocatable **parm::pot\_orgp**  
*amount of organic P in pothole water body (kg P)*
- real \*8, dimension(:), allocatable **parm::pot\_orgn**  
*amount of organic N in pothole water body (kg N)*
- real \*8, dimension(:), allocatable **parm::pot\_mps**  
*amount of stable mineral pool P in pothole water body (kg N)*
- real \*8, dimension(:), allocatable **parm::pot\_mpa**  
*amount of active mineral pool P in pothole water body (kg N)*
- real \*8, dimension(:), allocatable **parm::tile\_solpo**
- integer **parm::ia\_b**  
*print ascii or binary files (none)*
- integer **parm::ihumus**  
*ihumus = 0 do no print file*  
*ihumus = 1 print output.wql*
- integer **parm::itemp**
- integer **parm::isnow**
- integer, dimension(46) **parm::ipdvar**  
*output variable codes for output.rch file (none)*
- integer, dimension(mhruo) **parm::ipdvas**  
*output variable codes for output.hru file (none)*
- integer, dimension(msubo) **parm::ipdvab**  
*output variable codes for output.sub file (none)*
- integer, dimension(:), allocatable **parm::ipdhru**  
*HRUs whose output information will be printed to the output.hru and output.wtr files.*
- real \*8, dimension(mstdo) **parm::wshddayo**  
*watershed daily output array (varies)*  
*wshddayo(1) average amountof precipitation in watershed for the day (mm H2O)*  
*wshddayo(3) surface runoff in watershed for day (mm H2O)*  
*wshddayo(4) lateral flow contribution to streamflow in watershed for day (mm H2O)*  
*wshddayo(5) water percolation past bottom of soil profile in watershed for day (mm H2O)*  
*wshddayo(6) water yield to streamflow from HRUs in watershed for day (mm H2O)*  
*wshddayo(7) actual evapotranspiration in watershed for day (mm H2O)*  
*wshddayo(8) average maximum temperature in watershed for the day (deg C)*  
*wshddayo(9) average minimum temperature in watershed for the day (deg C)*  
*wshddayo(11) net change in sediment of reservoirs in watershed for day (metric tons)*  
*wshddayo(12) sediment yield from HRUs in watershed for day (metric tons or metric tons/ha)*  
*wshddayo(13) sediment loading to ponds in watershed for day (metric tons)*  
*wshddayo(14) sediment loading from ponds in watershed for day (metric tons)*  
*wshddayo(15) net change in sediment level in ponds in watershed for day (metric tons)*  
*wshddayo(16) sediment loading to wetlands for day in watershed (metric tons)*  
*wshddayo(17) sediment loading to main channels from wetlands for day in watershed (metric tons)*  
*wshddayo(18) net change in sediment in wetlands for day in watershed (metric tons)*  
*wshddayo(19) evaporation from ponds in watershed for day (m<sup>3</sup> H2O)*  
*wshddayo(20) seepage from ponds in watershed for day (m<sup>3</sup> H2O)*  
*wshddayo(21) precipitation on ponds in watershed for day (m<sup>3</sup> H2O)*  
*wshddayo(22) volume of water entering ponds in watershed for day (m<sup>3</sup> H2O)*  
*wshddayo(23) volume of water leaving ponds in watershed for day (m<sup>3</sup> H2O)*  
*wshddayo(24) evaporation from wetlands for day in watershed (m<sup>3</sup> H2O)*  
*wshddayo(25) seepage from wetlands for day in watershed (m<sup>3</sup> H2O)*  
*wshddayo(26) precipitation on wetlands for day in watershed (m<sup>3</sup> H2O)*  
*wshddayo(27) volume of water entering wetlands on day in watershed (m<sup>3</sup> H2O)*  
*wshddayo(28) volume of water leaving wetlands on day in watershed (m<sup>3</sup> H2O)*  
*wshddayo(33) net change in water volume of ponds in watershed for day (m<sup>3</sup> H2O)*  
*wshddayo(34) net change in water volume of reservoirs in watershed for day (m<sup>3</sup> H2O)*

- wshddayo(35)* amount of water stored in soil profile in watershed at end of day (mm H2O)
- wshddayo(36)* snow melt in watershed for day (mm H2O)
- wshddayo(37)* sublimation in watershed for day (mm H2O)
- wshddayo(38)* average amount of tributary channel transmission losses in watershed on day (mm H2O)
- wshddayo(39)* freezing rain/snow fall in watershed for day (mm H2O)
- wshddayo(40)* organic N loading to stream in watershed for day (kg N/ha)
- wshddayo(41)* organic P loading to stream in watershed for day (kg P/ha)
- wshddayo(42)* nitrate loading to stream in surface runoff in watershed for day (kg N/ha)
- wshddayo(43)* soluble P loading to stream in watershed for day (kg P/ha)
- wshddayo(44)* plant uptake of N in watershed for day (kg N/ha)
- wshddayo(45)* nitrate loading to stream in lateral flow in watershed for day (kg N/ha)
- wshddayo(46)* nitrate percolation past bottom of soil profile in watershed for day (kg N/ha)
- wshddayo(104)* groundwater contribution to stream in watershed on day (mm H2O)
- wshddayo(105)* amount of water moving from shallow aquifer to plants/soil profile in watershed on day (mm H2O)
- wshddayo(106)* deep aquifer recharge in watershed on day (mm H2O)
- wshddayo(107)* total amount of water entering both aquifers in watershed on day (mm H2O)
- wshddayo(108)* potential evapotranspiration in watershed on day (mm H2O)
- wshddayo(109)* drainage tile flow contribution to stream in watershed on day (mm H2O)
- wshddayo(110)* NO3 yield (gwq) (kg/ha)
- wshddayo(111)* NO3 yield (tile) (mm H2O)
- real \*8, dimension(mstdo) [parm::wshdmono](#)
  - watershed monthly output array (see definitions for wshddayo array elements) (varies)*
  - wshdmono(1)* average amount of precipitation in watershed for the month (mm H2O)
  - wshdmono(3)* surface runoff in watershed for month (mm H2O)
  - wshdmono(4)* lateral flow contribution to streamflow in watershed for month (mm H2O)
  - wshdmono(5)* water percolation past bottom of soil profile in watershed for month (mm H2O)
  - wshdmono(6)* water yield to streamflow from HRUs in watershed for month (mm H2O)
  - wshdmono(7)* actual evapotranspiration in watershed for month (mm H2O)
  - wshdmono(8)* average maximum temperature in watershed for the month (deg C)
  - wshdmono(9)* average minimum temperature in watershed for the month (deg C)
  - wshdmono(12)* sediment yield from HRUs in watershed for the month (metric tons)
  - wshdmono(39)* freezing rain/snow fall in watershed for the month (mm H2O)
  - wshdmono(40)* organic N loading to stream in watershed for the month (kg N/ha)
  - wshdmono(41)* organic P loading to stream in watershed for the month (kg P/ha)
  - wshdmono(42)* nitrate loading to stream in surface runoff in watershed for the month (kg N/ha)
  - wshdmono(43)* soluble P loading to stream in watershed for the month (kg P/ha)
  - wshdmono(44)* plant uptake of N in watershed for the month (kg N/ha)
  - wshdmono(45)* nitrate loading to stream in lateral flow in watershed for the month (kg N/ha)
  - wshdmono(46)* nitrate percolation past bottom of soil profile in watershed for the month (kg N/ha)
  - wshdmono(104)* groundwater contribution to stream in watershed for the month (mm H2O)
  - wshdmono(108)* potential evapotranspiration in watershed for the month (mm H2O)
  - wshdmono(109)* drainage tile flow contribution to stream in watershed for the month (mm H2O)
- real \*8, dimension(mstdo) [parm::wshdyro](#)
  - watershed annual output array (varies)*
  - wshdyro(1)* average amount of precipitation in watershed for the year (mm H2O)
  - wshdyro(3)* surface runoff in watershed for year (mm H2O)
  - wshdyro(4)* lateral flow contribution to streamflow in watershed for year (mm H2O)
  - wshdyro(5)* water percolation past bottom of soil profile in watershed for year (mm H2O)
  - wshdyro(6)* water yield to streamflow from HRUs in watershed for year (mm H2O)
  - wshdyro(7)* actual evapotranspiration in watershed for year (mm H2O)
  - wshdyro(8)* average maximum temperature in watershed for the year (deg C)
  - wshdyro(9)* average minimum temperature in watershed for the year (deg C)
  - wshdyro(12)* sediment yield from HRUs in watershed for the year (metric tons)
  - wshdyro(40)* organic N loading to stream in watershed for the year (kg N/ha)
  - wshdyro(41)* organic P loading to stream in watershed for the year (kg P/ha)
  - wshdyro(42)* nitrate loading to stream in surface runoff in watershed for the year (kg N/ha)
  - wshdyro(43)* soluble P loading to stream in watershed for the year (kg P/ha)
  - wshdyro(44)* plant uptake of N in watershed for the year
  - wshdyro(45)* nitrate loading to stream in lateral flow in watershed for the year (kg N/ha)
  - wshdyro(46)* nitrate percolation past bottom of soil profile in watershed for the year (kg N/ha)
  - wshdyro(104)* groundwater contribution to stream in watershed for the year (mm H2O)
  - wshdyro(108)* potential evapotranspiration in watershed for the year (mm H2O)
  - wshdyro(109)* drainage tile flow contribution to stream in watershed for the year (mm H2O)

- real \*8, dimension(16) **parm::fcstaa0**
- real \*8, dimension(mstdo) **parm::wshdaao**
  - watershed average annual output array (varies)*
  - wshdaao(1) precipitation in watershed (mm H2O)*
  - wshdaao(3) surface runoff loading to main channel in watershed (mm H2O)*
  - wshdaao(4) lateral flow loading to main channel in watershed (mm H2O)*
  - wshdaao(5) percolation of water out of root zone in watershed (mm H2O)*
  - wshdaao(6) water yield to streamflow from HRUs in watershed for simulation (mm H2O)*
  - wshdaao(7) actual evapotranspiration in watershed (mm H2O)*
  - wshdaao(11) net change in sediment of reservoirs in watershed during simulation (metric tons/ha)*
  - wshdaao(12) sediment yield from HRUs in watershed for the simulation (metric tons/ha)*
  - wshdaao(13) sediment loading to ponds in watershed during simulation (metric tons/ha)*
  - wshdaao(14) sediment loading from ponds in watershed during simulation (metric tons/ha)*
  - wshdaao(15) net change in sediment level in ponds in watershed during simulation (metric tons/ha)*
  - wshdaao(19) evaporation from ponds in watershed ( $m^3$  H2O)*
  - wshdaao(19) evaporation from ponds in watershed during simulation (mm H2O)*
  - wshdaao(20) seepage from ponds in watershed during simulation (mm H2O)*
  - wshdaao(21) precipitation on ponds in watershed during simulation (mm H2O)*
  - wshdaao(22) volume of water entering ponds in watershed during simulation (mm H2O)*
  - wshdaao(23) volume of water leaving ponds in watershed during simulation (mm H2O)*
  - wshdaao(33) net change in water volume of ponds in watershed during simulation (mm H2O)*
  - wshdaao(34) net change in water volume of reservoirs in watershed during simulation (mm H2O)*
  - wshdaao(36) snow melt in watershed for simulation (mm H2O)*
  - wshdaao(38) average amount of tributary channel transmission losses in watershed during simulation (mm H2O)*
  - wshdaao(39) freezing rain/snow fall in watershed for the simulation (mm H2O)*
  - wshdaao(40) organic N loading to stream in watershed for the simulation (kg N/ha)*
  - wshdaao(41) organic P loading to stream in watershed for the simulation (kg P/ha)*
  - wshdaao(42) nitrate loading to stream in surface runoff in watershed for the simulation (kg N/ha)*
  - wshdaao(43) soluble P loading to stream in watershed for the simulation (kg P/ha)*
  - wshdaao(44) plant uptake of N in watershed for the simulation (kg N/ha)*
  - wshdaao(45) nitrate loading to stream in lateral flow in watershed for the simulation (kg N/ha)*
  - wshdaao(46) nitrate percolation past bottom of soil profile in watershed for the simulation (kg N/ha)*
  - wshdaao(104) groundwater contribution to stream in watershed for the simulation (shallow aquifer) (mm H2O)*
  - wshdaao(105) amount of water moving from shallow aquifer to plants/soil profile in watershed during simulation (mm H2O)*
  - wshdaao(106) deep aquifer recharge in watershed during simulation (mm H2O)*
  - wshdaao(107) total amount of water entering both aquifers in watershed during simulation (mm H2O)*
  - wshdaao(108) potential evapotranspiration in watershed for the simulation (mm H2O)*
  - wshdaao(109) drainage tile flow contribution to stream in watershed for the simulation (mm H2O)*
  - wshdaao(113) groundwater contribution to stream in watershed for the simulation (deep aquifer) (mm H2O)*
- real \*8, dimension(:, :), allocatable **parm::wpstdayo**
  - watershed daily pesticide output array (varies)*
  - wpstdayo(1,:) amount of pesticide type in surface runoff contribution to stream in watershed on day (in solution) (mg pst/ha)*
  - wpstdayo(2,:) amount of pesticide type in surface runoff contribution to stream in watershed on day (sorbed to sediment) (mg pst/ha)*
  - wpstdayo(3,:) amount of pesticide type leached from soil profile in watershed on day (kg pst/ha)*
  - wpstdayo(4,:) amount of pesticide type in lateral flow contribution to stream in watershed on day (kg pst/ha)*
- real \*8, dimension(:, :), allocatable **parm::wpstmono**
- real \*8, dimension(:, :), allocatable **parm::wpstyro**
- real \*8, dimension(:, :), allocatable **parm::bio\_hv**
  - harvested biomass (dry weight) (kg/ha)*
- real \*8, dimension(:, :), allocatable **parm::yldkg**
  - yield (dry weight) by crop type in the HRU (kg/ha)*
- real \*8, dimension(:, :), allocatable **parm::rchmono**
  - reach monthly output array (varies)*
  - rchmono(1,:) flow into reach during month ( $m^3/s$ )*
  - rchmono(2,:) flow out of reach during month ( $m^3/s$ )*
  - rchmono(3,:) sediment transported into reach during month (metric tons)*
  - rchmono(4,:) sediment transported out of reach during month (metric tons)*
  - rchmono(5,:) sediment concentration in outflow during month (mg/L)*

- rchmono(6,:)* organic N transported into reach during month (kg N)
  - rchmono(7,:)* organic N transported out of reach during month (kg N)
  - rchmono(8,:)* organic P transported into reach during month (kg P)
  - rchmono(9,:)* organic P transported out of reach during month (kg P)
  - rchmono(10,:)* evaporation from reach during month ( $m^3/s$ )
  - rchmono(11,:)* transmission losses from reach during month ( $m^3/s$ )
  - rchmono(12,:)* conservative metal #1 transported out of reach during month (kg)
  - rchmono(13,:)* conservative metal #2 transported out of reach during month (kg)
  - rchmono(14,:)* conservative metal #3 transported out of reach during month (kg)
  - rchmono(15,:)* nitrate transported into reach during month (kg N)
  - rchmono(16,:)* nitrate transported out of reach during month (kg N)
  - rchmono(17,:)* soluble P transported into reach during month (kg P)
  - rchmono(18,:)* soluble P transported out of reach during month (kg P)
  - rchmono(19,:)* soluble pesticide transported into reach during month (mg pst)
  - rchmono(20,:)* soluble pesticide transported out of reach during month (mg pst)
  - rchmono(21,:)* sorbed pesticide transported into reach during month (mg pst)
  - rchmono(22,:)* sorbed pesticide transported out of reach during month (mg pst)
  - rchmono(23,:)* amount of pesticide lost through reactions in reach during month (mg pst)
  - rchmono(24,:)* amount of pesticide lost through volatilization from reach during month (mg pst)
  - rchmono(25,:)* amount of pesticide settling out of reach to bed sediment during month (mg pst)
  - rchmono(26,:)* amount of pesticide resuspended from bed sediment to reach during month (mg pst)
  - rchmono(27,:)* amount of pesticide diffusing from reach to bed sediment during month (mg pst)
  - rchmono(28,:)* amount of pesticide in sediment layer lost through reactions during month (mg pst)
  - rchmono(29,:)* amount of pesticide in sediment layer lost through burial during month (mg pst)
  - rchmono(30,:)* chlorophyll-a transported into reach during month (kg chla)
  - rchmono(31,:)* chlorophyll-a transported out of reach during month (kg chla)
  - rchmono(32,:)* ammonia transported into reach during month (kg N)
  - rchmono(33,:)* ammonia transported out of reach during month (kg N)
  - rchmono(34,:)* nitrite transported into reach during month (kg N)
  - rchmono(35,:)* nitrite transported out of reach during month (kg N)
  - rchmono(36,:)* CBOD transported into reach during month (kg O<sub>2</sub>)
  - rchmono(37,:)* CBOD transported out of reach during month (kg O<sub>2</sub>)
  - rchmono(38,:)* dissolved oxygen transported into reach during month (kg O<sub>2</sub>)
  - rchmono(39,:)* dissolved oxygen transported out of reach during month (kg O<sub>2</sub>)
  - rchmono(40,:)* persistent bacteria transported out of reach during month (kg bact)
  - rchmono(41,:)* less persistent bacteria transported out of reach during month (kg bact)
  - rchmono(43,:)* total N (org N + no<sub>3</sub> + no<sub>2</sub> + nh<sub>4</sub> outs) (kg)
  - rchmono(44,:)* total P (org P + sol p outs) (kg)
- real \*8, dimension(:,:), allocatable [parm::rchyro](#)
    - reach* annual output array (varies)
    - rchyro(1,:)* flow into reach during year ( $m^3/s$ )
    - rchyro(2,:)* flow out of reach during year ( $m^3/s$ )
    - rchyro(3,:)* sediment transported into reach during year (metric tons)
    - rchyro(4,:)* sediment transported out of reach during year (metric tons)
    - rchyro(5,:)* sediment concentration in outflow during year (mg/L)
    - rchyro(6,:)* organic N transported into reach during year (kg N)
    - rchyro(7,:)* organic N transported out of reach during year (kg N)
    - rchyro(8,:)* organic P transported into reach during year (kg P)
    - rchyro(9,:)* organic P transported out of reach during year (kg P)
    - rchyro(10,:)* evaporation from reach during year ( $m^3/s$ )
    - rchyro(11,:)* transmission losses from reach during year ( $m^3/s$ )
    - rchyro(12,:)* conservative metal #1 transported out of reach during year (kg)
    - rchyro(13,:)* conservative metal #2 transported out of reach during year (kg)
    - rchyro(14,:)* conservative metal #3 transported out of reach during year (kg)
    - rchyro(15,:)* nitrate transported into reach during year (kg N)
    - rchyro(16,:)* nitrate transported out of reach during year (kg N)
    - rchyro(17,:)* soluble P transported into reach during year (kg P)
    - rchyro(18,:)* soluble P transported out of reach during year (kg P)
    - rchyro(19,:)* soluble pesticide transported into reach during year (mg pst)
    - rchyro(20,:)* soluble pesticide transported out of reach during year (mg pst)
    - rchyro(21,:)* sorbed pesticide transported into reach during year (mg pst)
    - rchyro(22,:)* sorbed pesticide transported out of reach during year (mg pst)
    - rchyro(23,:)* amount of pesticide lost through reactions in reach during year!> (mg pst)



- rchyro(24,:)* amount of pesticide lost through volatilization from reach during year (mg pst)
- rchyro(25,:)* amount of pesticide settling out of reach to bed sediment during year (mg pst)
- rchyro(26,:)* amount of pesticide resuspended from bed sediment to reach during year (mg pst)
- rchyro(27,:)* amount of pesticide diffusing from reach to bed sediment during year (mg pst)
- rchyro(28,:)* amount of pesticide in sediment layer lost through reactions during year (mg pst)
- rchyro(29,:)* amount of pesticide in sediment layer lost through burial during year (mg pst)
- rchyro(30,:)* chlorophyll-a transported into reach during year (kg chla)
- rchyro(31,:)* chlorophyll-a transported out of reach during year (kg chla)
- rchyro(32,:)* ammonia transported into reach during year (kg N)
- rchyro(33,:)* ammonia transported out of reach during year (kg N)
- rchyro(34,:)* nitrite transported into reach during year (kg N)
- rchyro(35,:)* nitrite transported out of reach during year (kg N)
- rchyro(36,:)* CBOD transported into reach during year (kg O2)
- rchyro(37,:)* CBOD transported out of reach during year (kg O2)
- rchyro(38,:)* dissolved oxygen transported into reach during year (kg O2)
- rchyro(39,:)* dissolved oxygen transported out of reach during year (kg O2)
- rchyro(40,:)* persistent bacteria transported out of reach during year (kg bact)
- rchyro(41,:)* less persistent bacteria transported out of reach during year (kg bact)
- real \*8, dimension(:,:), allocatable [parm::wpstaaao](#)
  - wpstaaao(1,:)* amount of pesticide type in surface runoff contribution to stream in watershed (in solution) - average annual (mg pst/ha)
  - wpstaaao(2,:)* amount of pesticide type in surface runoff contribution to stream in watershed (sorbed to sediment) -average annual (mg pst/ha)
  - wpstaaao(3,:)* amount of pesticide type leached from soil profile in watershed - average annual (kg pst/ha)
  - wpstaaao(4,:)* amount of pesticide type in lateral flow contribution to stream in watershed - average annual (kg pst/ha)
- real \*8, dimension(:,:), allocatable [parm::hrumono](#)
  - HRU monthly output data array (varies)
  - hrumono(1,:)* precipitation in HRU during month (mm H2O)
  - hrumono(2,:)* amount of precipitation falling as freezing rain/snow in HRU during month (mm H2O)
  - hrumono(3,:)* amount of snow melt in HRU during month (mm H2O)
  - hrumono(4,:)* amount of surface runoff to main channel from HRU during month (ignores impact of transmission losses) (mm H2O)
  - hrumono(5,:)* amount of lateral flow contribution to main channel from HRU during month (mm H2O)
  - hrumono(6,:)* amount of groundwater flow contribution to main channel from HRU during month (mm H2O)
  - hrumono(7,:)* amount of water moving from shallow aquifer to plants or soil profile in HRU during month (mm H2O)
  - hrumono(8,:)* amount of water recharging deep aquifer in HRU during month (mm H2O)
  - hrumono(9,:)* total amount of water entering both aquifers from HRU during month (mm H2O)
  - hrumono(10,:)* water yield (total amount of water entering main channel) from HRU during month (mm H2O)
  - hrumono(11,:)* amount of water percolating out of the soil profile and into the vadose zone in HRU during month (mm H2O)
  - hrumono(12,:)* actual evapotranspiration in HRU during month (mm H2O)
  - hrumono(13,:)* amount of transmission losses from tributary channels in HRU for month (mm H2O)
  - hrumono(14,:)* sediment yield from HRU for month (metric tons/ha)
  - hrumono(15,:)* actual amount of transpiration that occurs during month in HRU (mm H2O)
  - hrumono(16,:)* actual amount of evaporation (from soil) that occurs during month in HRU (mm H2O)
  - hrumono(17,:)* amount of nitrogen applied in continuous fertilizer operation during month in HRU (kg N/ha)
  - hrumono(18,:)* amount of phosphorus applied in continuous fertilizer operation during month in HRU (kg P/ha)
  - hrumono(19,:)* amount of surface runoff generated during month in HRU (mm H2O)
  - hrumono(20,:)* CN values during month in HRU (none)
  - hrumono(21,:)* sum of daily soil water values used to calculate the curve number (mm H2O)
  - hrumono(22,:)* amount of irrigation water applied to HRU during month (mm H2O)
  - hrumono(23,:)* amount of water removed from shallow aquifer in HRU for irrigation during month (mm H2O)
  - hrumono(24,:)* amount of water removed from deep aquifer in HRU for irrigation during month (mm H2O)
  - hrumono(25,:)* potential evapotranspiration in HRU during month (mm H2O)
  - hrumono(26,:)* monthly amount of N (organic & mineral) applied in HRU during grazing (kg N/ha)
  - hrumono(27,:)* monthly amount of P (organic & mineral) applied in HRU during grazing (kg P/ha)
  - hrumono(28,:)* monthly amount of N (organic & mineral) auto-applied in HRU (kg N/ha)
  - hrumono(29,:)* monthly amount of P (organic & mineral) auto-applied in HRU (kg P/ha)
  - hrumono(30,:)* sum of daily soil temperature values (deg C)
  - hrumono(31,:)* water stress days in HRU during month (stress days)
  - hrumono(32,:)* temperature stress days in HRU during month (stress days)
  - hrumono(33,:)* nitrogen stress days in HRU during month (stress days)
  - hrumono(34,:)* phosphorus stress days in HRU during month (stress days)

- humono(35,:)* organic nitrogen in surface runoff in HRU during month (kg N/ha)
  - humono(36,:)* organic phosphorus in surface runoff in HRU during month (kg P/ha)
  - humono(37,:)* nitrate in surface runoff in HRU during month (kg N/ha)
  - humono(38,:)* nitrate in lateral flow in HRU during month (kg N/ha)
  - humono(39,:)* soluble phosphorus in surface runoff in HRU during month (kg P/ha)
  - humono(40,:)* amount of nitrogen removed from soil by plant uptake in HRU during month (kg N/ha)
  - humono(41,:)* nitrate percolating past bottom of soil profile in HRU during month (kg N/ha)
  - humono(42,:)* amount of phosphorus removed from soil by plant uptake in HRU during month (kg P/ha)
  - humono(43,:)* amount of phosphorus moving from labile mineral to active mineral pool in HRU during month (kg P/ha)
  - humono(44,:)* amount of phosphorus moving from active mineral to stable mineral pool in HRU during month (kg P/ha)
  - humono(45,:)* amount of nitrogen applied to HRU in fertilizer and grazing operations during month (kg N/ha)
  - humono(46,:)* amount of phosphorus applied to HRU in fertilizer and grazing operations during month (kg P/ha)
  - humono(47,:)* amount of nitrogen added to soil by fixation in HRU during month (kg N/ha)
  - humono(48,:)* amount of nitrogen lost by denitrification in HRU during month (kg N/ha)
  - humono(49,:)* amount of nitrogen moving from active organic to nitrate pool in HRU during month (kg N/ha)
  - humono(50,:)* amount of nitrogen moving from active organic to stable organic pool in HRU during month (kg N/ha)
  - humono(51,:)* amount of phosphorus moving from organic to labile mineral pool in HRU during month (kg P/ha)
  - humono(52,:)* amount of nitrogen moving from fresh organic to nitrate and active organic pools in HRU during month (kg N/ha)
  - humono(53,:)* amount of phosphorus moving from fresh organic to the labile mineral and organic pools in HRU during month (kg P/ha)
  - humono(54,:)* amount of nitrogen added to soil in rain (kg N/ha)
  - humono(61,:)* daily soil loss predicted with USLE equation (metric tons/ha)
  - humono(62,:)* drainage tile flow contribution to main channel from HRU in month (mm H<sub>2</sub>O)
  - humono(63,:)* less persistent bacteria transported to main channel from HRU during month (bacteria/ha)
  - humono(64,:)* persistent bacteria transported to main channel from HRU during month (bacteria/ha)
  - humono(65,:)* nitrate loading from groundwater in HRU to main channel during month (kg N/ha)
  - humono(66,:)* soluble P loading from groundwater in HRU to main channel during month (kg P/ha)
  - humono(67,:)* loading of mineral P attached to sediment in HRU to main channel during month (kg P/ha)
- real \*8, dimension(:,:), allocatable [parm::rchdy](#)
    - daily reach output array (varies)*
    - rchdy(1,:)* flow into reach on day (m<sup>3</sup>/s)
    - rchdy(2,:)* flow out of reach on day (m<sup>3</sup>/s)
    - rchdy(3,:)* evaporation from reach on day (m<sup>3</sup>/s)
    - rchdy(4,:)* transmission losses from reach on day (m<sup>3</sup>/s)
    - rchdy(5,:)* sediment transported into reach on day (metric tons)
    - rchdy(6,:)* sediment transported out of reach on day (metric tons)
    - rchdy(7,:)* sediment concentration in outflow (mg/L)
    - rchdy(8,:)* organic N transported into reach on day (kg N)
    - rchdy(9,:)* organic N transported out of reach on day (kg N)
    - rchdy(10,:)* organic P transported into reach on day (kg P)
    - rchdy(11,:)* organic P transported out of reach on day (kg P)
    - rchdy(12,:)* nitrate transported into reach on day (kg N)
    - rchdy(13,:)* nitrate transported out of reach on day (kg N)
    - rchdy(14,:)* ammonia transported into reach on day (kg N)
    - rchdy(15,:)* ammonia transported out of reach on day (kg N)
    - rchdy(16,:)* nitrite transported into reach on day (kg N)
    - rchdy(17,:)* nitrite transported out of reach on day (kg N)
    - rchdy(18,:)* soluble P transported into reach on day (kg P)
    - rchdy(19,:)* soluble P transported out of reach on day (kg P)
    - rchdy(20,:)* chlorophyll-a transported into reach on day (kg chl<sub>a</sub>)
    - rchdy(21,:)* chlorophyll-a transported out of reach on day (kg chl<sub>a</sub>)
    - rchdy(22,:)* CBOD transported into reach on day (kg O<sub>2</sub>)
    - rchdy(23,:)* CBOD transported out of reach on day (kg O<sub>2</sub>)
    - rchdy(24,:)* dissolved oxygen transported into reach on day (kg O<sub>2</sub>)
    - rchdy(25,:)* dissolved oxygen transported out of reach on day (kg O<sub>2</sub>)
    - rchdy(26,:)* soluble pesticide transported into reach on day (mg pst)
    - rchdy(27,:)* soluble pesticide transported out of reach on day (mg pst)
    - rchdy(28,:)* sorbed pesticide transported into reach on day (mg pst)
    - rchdy(29,:)* sorbed pesticide transported out of reach on day (mg pst)
    - rchdy(30,:)* amount of pesticide lost through reactions in reach on day (mg pst)

*rchdy(31,:)* amount of pesticide lost through volatilization from reach on day (mg pst)  
*rchdy(32,:)* amount of pesticide settling out of reach to bed sediment on day (mg pst)  
*rchdy(33,:)* amount of pesticide resuspended from bed sediment to reach on day (mg pst)  
*rchdy(34,:)* amount of pesticide diffusing from reach to bed sediment on day (mg pst)  
*rchdy(35,:)* amount of pesticide in sediment layer lost through reactions on day (mg pst)  
*rchdy(36,:)* amount of pesticide in sediment layer lost through burial on day (mg pst)  
*rchdy(37,:)* amount of pesticide stored in river bed sediments (mg pst)  
*rchdy(38,:)* persistent bacteria transported out of reach on day (kg bact)  
*rchdy(39,:)* less persistent bacteria transported out of reach on day (kg bact)  
*rchdy(40,:)* amount of conservative metal #1 transported out of reach on day (kg)  
*rchdy(41,:)* amount of conservative metal #2 transported out of reach on day (kg)  
*rchdy(42,:)* amount of conservative metal #3 transported out of reach on day (kg)  
*rchdy(43,:)* total N (org N + no3 + no2 + nh4 outs) (kg)  
*rchdy(44,:)* total P (org P + sol p outs) (kg)

- real \*8, dimension(:,:), allocatable **parm::hruyro**

*HRU* annual output array (varies) *hruyro(1,:)* precipitation in HRU during year (mm H2O)  
*hruyro(2,:)* amount of precipitation falling as freezing rain/snow in HRU during year (mm H2O)  
*hruyro(3,:)* amount of snow melt in HRU during year (mm H2O)  
*hruyro(4,:)* amount of surface runoff to main channel from HRU during year (ignores impact of transmission losses) (mm H2O)  
*hruyro(5,:)* amount of lateral flow contribution to main channel from HRU during year (mm H2O)  
*hruyro(6,:)* amount of groundwater flow contribution to main channel from HRU during year (mm H2O)  
*hruyro(7,:)* amount of water moving from shallow aquifer to plants or soil profile in HRU during year (mm H2O)  
*hruyro(8,:)* amount of water recharging deep aquifer in HRU during year (mm H2O)  
*hruyro(9,:)* total amount of water entering both aquifers from HRU during year (mm H2O)  
*hruyro(10,:)* water yield (total amount of water entering main channel) from HRU during year (mm H2O)  
*hruyro(11,:)* amount of water percolating out of the soil profile and into the vadose zone in HRU during year (mm H2O)  
*hruyro(12,:)* actual evapotranspiration in HRU during year (mm H2O)  
*hruyro(13,:)* amount of transmission losses from tributary channels in HRU for year (mm H2O)  
*hruyro(14,:)* sediment yield from HRU for year (metric tons/ha)  
*hruyro(15,:)* actual amount of transpiration that occurs during year in HRU (mm H2O)  
*hruyro(16,:)* actual amount of evaporation (from soil) that occurs during year in HRU (mm H2O)  
*hruyro(17,:)* amount of nitrogen applied in continuous fertilizer operation during year in HRU (kg N/ha)  
*hruyro(18,:)* amount of phosphorus applied in continuous fertilizer operation during year in HRU (kg P/ha)  
*hruyro(23,:)* amount of water removed from shallow aquifer in HRU for irrigation during year (mm H2O)  
*hruyro(24,:)* amount of water removed from deep aquifer in HRU for irrigation during year (mm H2O)  
*hruyro(25,:)* potential evapotranspiration in HRU during year (mm H2O)  
*hruyro(26,:)* annual amount of N (organic & mineral) applied in HRU during grazing (kg N/ha)  
*hruyro(27,:)* annual amount of P (organic & mineral) applied in HRU during grazing (kg P/ha)  
*hruyro(28,:)* annual amount of N (organic & mineral) auto-applied in HRU (kg N/ha)  
*hruyro(29,:)* annual amount of P (organic & mineral) auto-applied in HRU (kg P/ha)  
*hruyro(31,:)* water stress days in HRU during year (stress days)  
*hruyro(32,:)* temperature stress days in HRU during year (stress days)  
*hruyro(33,:)* nitrogen stress days in HRU during year (stress days)  
*hruyro(34,:)* phosphorus stress days in HRU during year (stress days)  
*hruyro(35,:)* organic nitrogen in surface runoff in HRU during year (kg N/ha)  
*hruyro(36,:)* organic phosphorus in surface runoff in HRU during year (kg P/ha)  
*hruyro(37,:)* nitrate in surface runoff in HRU during year (kg N/ha)  
*hruyro(38,:)* nitrate in lateral flow in HRU during year (kg N/ha)  
*hruyro(39,:)* soluble phosphorus in surface runoff in HRU during year (kg P/ha)  
*hruyro(40,:)* amount of nitrogen removed from soil by plant uptake in HRU during year (kg N/ha)  
*hruyro(41,:)* nitrate percolating past bottom of soil profile in HRU during year (kg N/ha)  
*hruyro(42,:)* amount of phosphorus removed from soil by plant uptake in HRU during year (kg P/ha)  
*hruyro(43,:)* amount of phosphorus moving from labile mineral to active mineral pool in HRU during year (kg P/ha)  
*hruyro(44,:)* amount of phosphorus moving from active mineral to stable mineral pool in HRU during year (kg P/ha)  
*hruyro(45,:)* amount of nitrogen applied to HRU in fertilizer and grazing operations during year (kg N/ha)  
*hruyro(46,:)* amount of phosphorus applied to HRU in fertilizer and grazing operations during year (kg P/ha)  
*hruyro(47,:)* amount of nitrogen added to soil by fixation in HRU during year (kg N/ha)  
*hruyro(48,:)* amount of nitrogen lost by denitrification in HRU during year (kg N/ha)  
*hruyro(49,:)* amount of nitrogen moving from active organic to nitrate pool in HRU during year (kg N/ha)  
*hruyro(50,:)* amount of nitrogen moving from active organic to stable organic pool in HRU during year (kg N/ha)  
*hruyro(51,:)* amount of phosphorus moving from organic to labile mineral pool in HRU during year (kg P/ha)



- hruyro(52,:)* amount of nitrogen moving from fresh organic to nitrate and active organic pools in HRU during year (kg N/ha)
- hruyro(53,:)* amount of phosphorus moving from fresh organic to the labile mineral and organic pools in HRU during year (kg P/ha)
- hruyro(54,:)* amount of nitrogen added to soil in rain during year (kg N/ha)
- hruyro(61,:)* daily soil loss predicted with USLE equation (metric tons/ha)
- hruyro(63,:)* less persistent bacteria transported to main channel from HRU during year (# bacteria/ha)
- hruyro(64,:)* persistent bacteria transported to main channel from HRU during year (# bacteria/ha)
- hruyro(65,:)* nitrate loading from groundwater in HRU to main channel during year (kg N/ha)
- hruyro(66,:)* soluble P loading from groundwater in HRU to main channel during year (kg P/ha)
- hruyro(67,:)* loading of mineral P attached to sediment in HRU to main channel during year (kg P/ha)
- real \*8, dimension(:,:), allocatable [parm::rchaa0](#)
  - reach average annual output array (varies)
  - rchaa0(1,:)* flow into reach during simulation ( $m^3/s$ )
  - rchaa0(2,:)* flow out of reach during simulation ( $m^3/s$ )
  - rchaa0(3,:)* sediment transported into reach during simulation (metric tons)
  - rchaa0(4,:)* sediment transported out of reach during simulation (metric tons)
  - rchaa0(5,:)* sediment concentration in outflow during simulation (mg/L)
  - rchaa0(6,:)* organic N transported into reach during simulation (kg N)
  - rchaa0(7,:)* organic N transported out of reach during simulation (kg N)
  - rchaa0(8,:)* organic P transported into reach during simulation (kg P)
  - rchaa0(9,:)* organic P transported out of reach during simulation (kg P)
  - rchaa0(10,:)* evaporation from reach during simulation ( $m^3/s$ )
  - rchaa0(11,:)* transmission losses from reach during simulation ( $m^3/s$ )
  - rchaa0(12,:)* conservative metal #1 transported out of reach during simulation (kg)
  - rchaa0(13,:)* conservative metal #2 transported out of reach during simulation (kg)
  - rchaa0(14,:)* conservative metal #3 transported out of reach during simulation (kg)
  - rchaa0(15,:)* nitrate transported into reach during simulation (kg N)
  - rchaa0(16,:)* nitrate transported out of reach during simulation (kg N)
  - rchaa0(17,:)* soluble P transported into reach during simulation (kg P)
  - rchaa0(18,:)* soluble P transported out of reach during simulation (kg P)
  - rchaa0(19,:)* soluble pesticide transported into reach during simulation
  - rchaa0(20,:)* soluble pesticide transported out of reach during simulation
  - rchaa0(21,:)* sorbed pesticide transported into reach during simulation
  - rchaa0(22,:)* sorbed pesticide transported out of reach during simulation
  - rchaa0(23,:)* amount of pesticide lost through reactions in reach during simulation
  - rchaa0(24,:)* amount of pesticide lost through volatilization from reach during simulation
  - rchaa0(25,:)* amount of pesticide settling out of reach to bed sediment during simulation
  - rchaa0(26,:)* amount of pesticide resuspended from bed sediment to reach during simulation
  - rchaa0(27,:)* amount of pesticide diffusing from reach to bed sediment during simulation
  - rchaa0(28,:)* amount of pesticide in sediment layer lost through reactions during simulation
  - rchaa0(29,:)* amount of pesticide in sediment layer lost through burial during simulation
  - rchaa0(30,:)* chlorophyll-a transported into reach during simulation (kg chl-a)
  - rchaa0(31,:)* chlorophyll-a transported out of reach during simulation (kg chl-a)
  - rchaa0(32,:)* ammonia transported into reach during simulation (kg N)
  - rchaa0(33,:)* ammonia transported out of reach during simulation (kg N)
  - rchaa0(34,:)* nitrite transported into reach during simulation (kg N)
  - rchaa0(35,:)* nitrite transported out of reach during simulation (kg N)
  - rchaa0(36,:)* CBOD transported into reach during simulation (kg O<sub>2</sub>)
  - rchaa0(37,:)* CBOD transported out of reach during simulation (kg O<sub>2</sub>)
  - rchaa0(38,:)* dissolved oxygen transported into reach during simulation (kg O<sub>2</sub>)
  - rchaa0(39,:)* dissolved oxygen transported out of reach during simulation (kg O<sub>2</sub>)
  - rchaa0(40,:)* persistent bacteria transported out of reach during simulation (kg bact)
  - rchaa0(41,:)* less persistent bacteria transported out of reach during simulation (kg bact)
  - rchaa0(43,:)* Total N (org N + no<sub>3</sub> + no<sub>2</sub> + nh<sub>4</sub> outs) (kg)
  - rchaa0(44,:)* Total P (org P + sol p outs) (kg)
- real \*8, dimension(:,:), allocatable [parm::submono](#)
  - subbasin monthly output array (varies)
  - submono(1,:)* precipitation in subbasin for month (mm H<sub>2</sub>O)
  - submono(2,:)* snow melt in subbasin for month (mm H<sub>2</sub>O)
  - submono(3,:)* surface runoff loading in subbasin for month (mm H<sub>2</sub>O)
  - submono(4,:)* water yield from subbasin for month (mm H<sub>2</sub>O)
  - submono(5,:)* potential evapotranspiration in subbasin for month (mm H<sub>2</sub>O)

- submono(6,:)* actual evapotranspiration in subbasin for month (mm H2O)
  - submono(7,:)* sediment yield from subbasin for month (metric tons/ha)
  - submono(8,:)* organic N loading from subbasin for month (kg N/ha)
  - submono(9,:)* organic P loading from subbasin for month (kg P/ha)
  - submono(10,:)* NO3 loading from surface runoff in subbasin for month (kg N/ha)
  - submono(11,:)* soluble P loading from subbasin for month (kg P/ha)
  - submono(12,:)* groundwater loading from subbasin for month (mm H2O)
  - submono(13,:)* percolation out of soil profile in subbasin for month (mm H2O)
  - submono(14,:)* loading to reach of mineral P attached to sediment from subbasin for month (kg P/ha)
- real \*8, dimension(:,:), allocatable **parm::subyro**
  - subbasin annual output array (varies)*
  - subyro(1,:)* precipitation in subbasin for year (mm H2O)
  - subyro(2,:)* snow melt in subbasin for year (mm H2O)
  - subyro(3,:)* surface runoff loading in subbasin for year (mm H2O)
  - subyro(4,:)* water yield from subbasin for year (mm H2O)
  - subyro(5,:)* potential evapotranspiration in subbasin for year (mm H2O)
  - subyro(6,:)* actual evapotranspiration in subbasin for year (mm H2O)
  - subyro(7,:)* sediment yield from subbasin for year (metric tons/ha)
  - subyro(8,:)* organic N loading from subbasin for year (kg N/ha)
  - subyro(9,:)* organic P loading from subbasin for year (kg P/ha)
  - subyro(10,:)* NO3 loading from surface runoff in subbasin for year (kg N/ha)
  - subyro(11,:)* soluble P loading from subbasin for year (kg P/ha)
  - subyro(12,:)* groundwater loading from subbasin for year (mm H2O)
  - subyro(13,:)* percolation out of soil profile in subbasin for year (mm H2O)
  - subyro(14,:)* loading to reach of mineral P attached to sediment from subbasin for year (kg P/ha)
- real \*8, dimension(:,:), allocatable **parm::hruaao**
  - HRU average annual output array (varies)*
  - hruaao(1,:)* precipitation in HRU during simulation (mm H2O)
  - hruaao(2,:)* amount of precipitation falling as freezing rain/snow in HRU during simulation (mm H2O)
  - hruaao(3,:)* amount of snow melt in HRU during simulation (mm H2O)
  - hruaao(4,:)* amount of surface runoff to main channel from HRU during simulation (ignores impact of transmission losses) (mm H2O)
  - hruaao(5,:)* amount of lateral flow contribution to main channel from HRU during simulation (mm H2O)
  - hruaao(6,:)* amount of groundwater flow contribution to main channel from HRU during simulation (mm H2O)
  - hruaao(7,:)* amount of water moving from shallow aquifer to plants or soil profile in HRU during simulation (mm H2O)
  - hruaao(8,:)* amount of water recharging deep aquifer in HRU during simulation (mm H2O)
  - hruaao(9,:)* total amount of water entering both aquifers from HRU during simulation (mm H2O)
  - hruaao(10,:)* water yield (total amount of water entering main channel) from HRU during simulation (mm H2O)
  - hruaao(11,:)* amount of water percolating out of the soil profile and into the vadose zone in HRU during simulation (mm H2O)
  - hruaao(12,:)* actual evapotranspiration in HRU during simulation
  - hruaao(13,:)* amount of transmission losses from tributary channels in HRU for simulation (mm H2O)
  - hruaao(14,:)* sediment yield from HRU for simulation (metric tons/ha)
  - hruaao(15,:)* actual amount of transpiration that occurs during simulation in HRU (mm H2O)
  - hruaao(16,:)* actual amount of evaporation (from soil) that occurs during simulation in HRU (mm H2O)
  - hruaao(17,:)* amount of nitrogen applied in continuous fertilizer operation in HRU for simulation (kg N/ha)
  - hruaao(18,:)* amount of phosphorus applied in continuous fertilizer operation in HRU for simulation (kg P/ha)
  - hruaao(22,:)* amount of irrigation water applied to HRU during simulation (mm H2O)
  - hruaao(23,:)* amount of water removed from shallow aquifer in HRU for irrigation during simulation (mm H2O)
  - hruaao(24,:)* amount of water removed from deep aquifer in HRU for irrigation during simulation (mm H2O)
  - hruaao(25,:)* potential evapotranspiration in HRU during simulation (mm H2O)
  - hruaao(26,:)* annual amount of N (organic & mineral) applied in HRU during grazing (kg N/ha)
  - hruaao(27,:)* annual amount of P (organic & mineral) applied in HRU during grazing (kg P/ha)
  - hruaao(28,:)* average annual amount of N (organic & mineral) auto-applied in HRU (kg N/ha)
  - hruaao(29,:)* average annual amount of P (organic & mineral) auto-applied in HRU (kg P/ha)
  - hruaao(31,:)* water stress days in HRU during simulation (stress days)
  - hruaao(32,:)* temperature stress days in HRU during simulation (stress days)
  - hruaao(33,:)* nitrogen stress days in HRU during simulation (stress days)
  - hruaao(34,:)* phosphorus stress days in HRU during simulation (stress days)
  - hruaao(35,:)* organic nitrogen in surface runoff in HRU during simulation (kg N/ha)
  - hruaao(36,:)* organic phosphorus in surface runoff in HRU during simulation (kg P/ha)
  - hruaao(37,:)* nitrate in surface runoff in HRU during simulation (kg N/ha)
  - hruaao(38,:)* nitrate in lateral flow in HRU during simulation (kg N/ha)

- hruaao(39,:) soluble phosphorus in surface runoff in HRU during simulation (kg P/ha)*
- hruaao(40,:) amount of nitrogen removed from soil by plant uptake in HRU during simulation (kg N/ha)*
- hruaao(41,:) nitrate percolating past bottom of soil profile in HRU during simulation (kg N/ha)*
- hruaao(42,:) amount of phosphorus removed from soil by plant uptake in HRU during simulation (kg P/ha)*
- hruaao(43,:) amount of phosphorus moving from labile mineral to active mineral pool in HRU during simulation (kg P/ha)*
- hruaao(44,:) amount of phosphorus moving from active mineral to stable mineral pool in HRU during simulation (kg P/ha)*
- hruaao(45,:) amount of nitrogen applied to HRU in fertilizer and grazing operations during simulation (kg N/ha)*
- hruaao(46,:) amount of phosphorus applied to HRU in fertilizer and grazing operations during simulation (kg P/ha)*
- hruaao(47,:) amount of nitrogen added to soil by fixation in HRU during simulation (kg N/ha)*
- hruaao(48,:) amount of nitrogen lost by denitrification in HRU during simulation (kg N/ha)*
- hruaao(49,:) amount of nitrogen moving from active organic to nitrate pool in HRU during simulation (kg N/ha)*
- hruaao(50,:) amount of nitrogen moving from active organic to stable organic pool in HRU during simulation (kg N/ha)*
- hruaao(51,:) amount of phosphorus moving from organic to labile mineral pool in HRU during simulation (kg P/ha)*
- hruaao(52,:) amount of nitrogen moving from fresh organic to nitrate and active organic pools in HRU during simulation (kg N/ha)*
- hruaao(53,:) amount of phosphorus moving from fresh organic to the labile mineral and organic pools in HRU during simulation (kg P/ha)*
- hruaao(54,:) amount of nitrogen added to soil in rain during simulation (kg N/ha)*
- hruaao(61,:) daily soil loss predicted with USLE equation (metric tons/ha)*
- hruaao(63,:) less persistent bacteria transported to main channel from HRU during simulation (# bacteria/ha)*
- hruaao(64,:) persistent bacteria transported to main channel from HRU during simulation (# bacteria/ha)*
- hruaao(65,:) nitrate loading from groundwater in HRU to main channel during simulation (kg N/ha)*
- hruaao(66,:) soluble P loading from groundwater in HRU to main channel during simulation (kg P/ha)*
- hruaao(67,:) loading of mineral P attached to sediment in HRU to main channel during simulation (kg P/ha)*
- real \*8, dimension(:,:), allocatable [parm::subbaao](#)
  - subbasin average annual output array (varies)*
  - subbaao(1,:) precipitation in subbasin for simulation (mm H2O)*
  - subbaao(2,:) snow melt in subbasin for simulation (mm H2O)*
  - subbaao(3,:) surface runoff loading in subbasin for simulation (mm H2O)*
  - subbaao(4,:) water yield from subbasin for simulation (mm H2O)*
  - subbaao(5,:) potential evapotranspiration in subbasin for simulation (mm H2O)*
  - subbaao(6,:) actual evapotranspiration in subbasin for simulation (mm H2O)*
  - subbaao(7,:) sediment yield from subbasin for simulation (metric tons/ha)*
  - subbaao(8,:) organic N loading from subbasin for simulation (kg N/ha)*
  - subbaao(9,:) organic P loading from subbasin for simulation (kg P/ha)*
  - subbaao(10,:) NO3 loading from surface runoff in subbasin for simulation (kg N/ha)*
  - subbaao(11,:) soluble P loading from subbasin for simulation (kg P/ha)*
  - subbaao(12,:) groundwater loading from subbasin for simulation (mm H2O)*
  - subbaao(13,:) percolation out of soil profile in subbasin for simulation (mm H2O)*
  - subbaao(14,:) loading to reach of mineral P attached to sediment from subbasin for simulation (kg P/ha)*
  - subbaao(18,i) groundwater?*
- real \*8, dimension(:,:), allocatable [parm::resoutm](#)
  - reservoir monthly output array (varies)*
  - resoutm(1,:) flow into reservoir during month ( $m^3/s$ )*
  - resoutm(2,:) flow out of reservoir during month ( $m^3/s$ )*
  - resoutm(3,:) sediment entering reservoir during month (metric tons)*
  - resoutm(4,:) sediment leaving reservoir during month (metric tons)*
  - resoutm(5,:) sediment concentration in reservoir during month (mg/L)*
  - resoutm(6,:) pesticide entering reservoir during month (mg pst)*
  - resoutm(7,:) pesticide lost from reservoir through reactions during month (mg pst)*
  - resoutm(8,:) pesticide lost from reservoir through volatilization during month (mg pst)*
  - resoutm(9,:) pesticide moving from water to sediment through settling during month (mg pst)*
  - resoutm(10,:) pesticide moving from sediment to water through resuspension during month (mg pst)*
  - resoutm(11,:) pesticide moving from water to sediment through diffusion during month (mg pst)*
  - resoutm(12,:) pesticide lost from reservoir sediment layer through reactions during month (mg pst)*
  - resoutm(13,:) pesticide lost from reservoir sediment layer through burial during month (mg pst)*
  - resoutm(14,:) pesticide transported out of reservoir during month (mg pst)*
  - resoutm(15,:) pesticide concentration in reservoir water during month (mg pst/ $m^3$ )*
  - resoutm(16,:) pesticide concentration in reservoir sediment layer during month (mg pst/ $m^3$ )*
  - resoutm(17,:) evaporation from reservoir during month ( $m^3 H_2O$ )*
  - resoutm(18,:) seepage from reservoir during month ( $m^3 H_2O$ )*

*resoutm(19,:)* precipitation on reservoir during month ( $\text{m}^3 \text{H}_2\text{O}$ )  
*resoutm(20,:)* water flowing into reservoir during month ( $\text{m}^3 \text{H}_2\text{O}$ )  
*resoutm(21,:)* water flowing out of reservoir during month ( $\text{m}^3 \text{H}_2\text{O}$ )  
*resoutm(22,:)* organic N entering reservoir during month (kg N)  
*resoutm(23,:)* organic N leaving reservoir during month (kg N)  
*resoutm(24,:)* organic P entering reservoir during month (kg P)  
*resoutm(25,:)* organic P leaving reservoir during month (kg P)  
*resoutm(26,:)* nitrate entering reservoir during month (kg N)  
*resoutm(27,:)* nitrate leaving reservoir during month (kg N)  
*resoutm(28,:)* nitrite entering reservoir during month (kg N)  
*resoutm(29,:)* nitrite leaving reservoir during month (kg N)  
*resoutm(30,:)* ammonia entering reservoir during month (kg N)  
*resoutm(31,:)* ammonia leaving reservoir during month (kg N)  
*resoutm(32,:)* mineral P entering reservoir during month (kg P)  
*resoutm(33,:)* mineral P leaving reservoir during month (kg P)  
*resoutm(34,:)* chlorophyll-a entering reservoir during month (kg chl-a)  
*resoutm(35,:)* chlorophyll-a leaving reservoir during month (kg chl-a)  
*resoutm(36,:)* organic P concentration in reservoir water during month (mg P/L)  
*resoutm(37,:)* mineral P concentration in reservoir water during month (mg P/L)  
*resoutm(38,:)* organic N concentration in reservoir water during month (mg N/L)  
*resoutm(39,:)* nitrate concentration in reservoir water during month (mg N/L)  
*resoutm(40,:)* nitrite concentration in reservoir water during month (mg N/L)  
*resoutm(41,:)* ammonia concentration in reservoir water during month (mg N/L)

- real \*8, dimension(:,:), allocatable [parm::resouty](#)

*reservoir annual output array (varies)*  
*resouty(1,:)* flow into reservoir during year ( $\text{m}^3/\text{s}$ )  
*resouty(2,:)* flow out of reservoir during year ( $\text{m}^3/\text{s}$ )  
*resouty(3,:)* sediment entering reservoir during year (metric tons)  
*resouty(4,:)* sediment leaving reservoir during year (metric tons)  
*resouty(5,:)* sediment concentration in reservoir during year (mg/L)  
*resouty(6,:)* pesticide entering reservoir during year (mg pst)  
*resouty(7,:)* pesticide lost from reservoir through reactions during year (mg pst)  
*resouty(8,:)* pesticide lost from reservoir through volatilization during year (mg pst)  
*resouty(9,:)* pesticide moving from water to sediment through settling during year (mg pst)  
*resouty(10,:)* pesticide moving from sediment to water through resuspension during year (mg pst)  
*resouty(11,:)* pesticide moving from water to sediment through diffusion during year (mg pst)  
*resouty(12,:)* pesticide lost from reservoir sediment layer through reactions during year (mg pst)  
*resouty(13,:)* pesticide lost from reservoir sediment layer through burial during year (mg pst)  
*resouty(14,:)* pesticide transported out of reservoir during year (mg pst)  
*resouty(15,:)* pesticide concentration in reservoir water during year (mg pst/ $\text{m}^3$ )  
*resouty(16,:)* pesticide concentration in reservoir sediment layer during year (mg pst/ $\text{m}^3$ )  
*resouty(17,:)* evaporation from reservoir during year ( $\text{m}^3 \text{H}_2\text{O}$ )  
*resouty(18,:)* seepage from reservoir during year ( $\text{m}^3 \text{H}_2\text{O}$ )  
*resouty(19,:)* precipitation on reservoir during year ( $\text{m}^3 \text{H}_2\text{O}$ )  
*resouty(22,:)* organic N entering reservoir during year (kg N)  
*resouty(23,:)* organic N leaving reservoir during year (kg N)  
*resouty(24,:)* organic P entering reservoir during year (kg P)  
*resouty(25,:)* organic P leaving reservoir during year (kg P)  
*resouty(26,:)* nitrate entering reservoir during year (kg N)  
*resouty(27,:)* nitrate leaving reservoir during year (kg N)  
*resouty(28,:)* nitrite entering reservoir during year (kg N)  
*resouty(29,:)* nitrite leaving reservoir during year (kg N)  
*resouty(30,:)* ammonia entering reservoir during year (kg N)  
*resouty(31,:)* ammonia leaving reservoir during year (kg N)  
*resouty(32,:)* mineral P entering reservoir during year (kg P)  
*resouty(33,:)* mineral P leaving reservoir during year (kg P)  
*resouty(34,:)* chlorophyll-a entering reservoir during year (kg chl-a)  
*resouty(35,:)* chlorophyll-a leaving reservoir during year (kg chl-a)  
*resouty(36,:)* organic P concentration in reservoir water during year (mg P/L)  
*resouty(37,:)* mineral P concentration in reservoir water during year (mg P/L)  
*resouty(38,:)* organic N concentration in reservoir water during year (mg N/L)  
*resouty(39,:)* nitrate concentration in reservoir water during year (mg N/L)  
*resouty(40,:)* nitrite concentration in reservoir water during year (mg N/L)



- resouty(41,:) ammonia concentration in reservoir water during year (mg N/L)*
- real \*8, dimension(:,:), allocatable [parm::resouta](#)
  - reservoir average annual output array (varies)*
  - resouta(3,:) sediment entering reservoir during simulation (metric tons)*
  - resouta(4,:) sediment leaving reservoir during simulation (metric tons)*
  - resouta(17,:) evaporation from reservoir during simulation ( $m^3 H_2O$ )*
  - resouta(18,:) seepage from reservoir during simulation ( $m^3 H_2O$ )*
  - resouta(19,:) precipitation on reservoir during simulation ( $m^3 H_2O$ )*
  - resouta(20,:) water entering reservoir during simulation ( $m^3 H_2O$ )*
  - resouta(21,:) water leaving reservoir during simulation ( $m^3 H_2O$ )*
- real \*8, dimension(12, 8) [parm::wshd\\_aamon](#)
  - array of watershed monthly average values (varies)*
  - wshd\_aamon(:,1) average annual precipitation in watershed falling during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,2) average annual freezing rain in watershed falling during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,3) average annual surface runoff in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,4) average annual lateral flow in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,5) average annual water yield in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,6) average annual actual evapotranspiration in watershed during month (mm H<sub>2</sub>O)*
  - wshd\_aamon(:,7) average annual sediment yield in watershed during month (metric tons)*
  - wshd\_aamon(:,8) average annual potential evapotranspiration in watershed during month (mm H<sub>2</sub>O)*
- real \*8, dimension(:,:), allocatable [parm::wtrmon](#)
  - HRU monthly output data array for impoundments (varies)*
  - wtrmon(1,:) evaporation from ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(2,:) seepage from ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(3,:) precipitation on ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(4,:) amount of water entering ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(5,:) sediment entering ponds in HRU for month (metric tons/ha)*
  - wtrmon(6,:) amount of water leaving ponds in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(7,:) sediment leaving ponds in HRU for month (metric tons/ha)*
  - wtrmon(8,:) precipitation on wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(9,:) volume of water entering wetlands from HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(10,:) sediment loading to wetlands for month from HRU (metric tons/ha)*
  - wtrmon(11,:) evaporation from wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(12,:) seepage from wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(13,:) volume of water leaving wetlands in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(14,:) sediment loading from wetlands in HRU to main channel during month (metric tons/ha)*
  - wtrmon(15,:) precipitation on potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(16,:) evaporation from potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(17,:) seepage from potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(18,:) water leaving potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(19,:) water entering potholes in HRU for month (mm H<sub>2</sub>O)*
  - wtrmon(20,:) sediment entering potholes in HRU for month (metric tons/ha)*
  - wtrmon(21,:) sediment leaving potholes in HRU for month (metric tons/ha)*
- real \*8, dimension(:,:), allocatable [parm::wtryr](#)
  - HRU impoundment annual output array (varies)*
  - wtryr(1,:) evaporation from ponds in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(2,:) seepage from ponds in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(3,:) precipitation on ponds in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(4,:) amount of water entering ponds in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(5,:) sediment entering ponds in HRU for year (metric tons/ha)*
  - wtryr(6,:) amount of water leaving ponds in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(7,:) sediment leaving ponds in HRU for year (metric tons/ha)*
  - wtryr(8,:) precipitation on wetlands in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(9,:) volume of water entering wetlands from HRU for year (mm H<sub>2</sub>O)*
  - wtryr(10,:) sediment loading to wetlands for year from HRU (metric tons/ha)*
  - wtryr(11,:) evaporation from wetlands in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(12,:) seepage from wetlands in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(13,:) volume of water leaving wetlands in HRU for year (mm H<sub>2</sub>O)*
  - wtryr(14,:) sediment loading from wetlands in HRU to main channel during year (metric tons/ha)*
  - wtryr(15,:) precipitation on potholes in HRU during year (mm H<sub>2</sub>O)*
  - wtryr(16,:) evaporation from potholes in HRU during year (mm H<sub>2</sub>O)*
  - wtryr(17,:) seepage from potholes in HRU during year (mm H<sub>2</sub>O)*

- wtryr(18,:)* water leaving potholes in HRU during year (mm H2O)
- wtryr(19,:)* water entering potholes in HRU during year (mm H2O)
- wtryr(20,:)* sediment entering potholes in HRU during year (metric tons/ha)
- wtryr(21,:)* sediment leaving potholes in HRU during year (metric tons/ha)
- real \*8, dimension(:,:), allocatable [parm::wtraa](#)
  - HRU impoundment average annual output array (varies)*
  - wtraa(1,:)* evaporation from ponds in HRU during simulation (mm H2O)
  - wtraa(2,:)* seepage from ponds in HRU during simulation (mm H2O)
  - wtraa(3,:)* precipitation on ponds in HRU during simulation (mm H2O)
  - wtraa(4,:)* amount of water entering ponds in HRU during simulation (mm H2O)
  - wtraa(5,:)* sediment entering ponds in HRU during simulation (metric tons/ha)
  - wtraa(6,:)* amount of water leaving ponds in HRU during simulation (mm H2O)
  - wtraa(7,:)* sediment leaving ponds in HRU during simulation (metric tons/ha)
  - wtraa(8,:)* precipitation on wetlands in HRU during simulation (mm H2O)
  - wtraa(9,:)* volume of water entering wetlands from HRU during simulation (mm H2O)
  - wtraa(10,:)* sediment loading to wetlands during simulation from HRU (metric tons/ha)
  - wtraa(11,:)* evaporation from wetlands in HRU during simulation (mm H2O)
  - wtraa(12,:)* seepage from wetlands in HRU during simulation (mm H2O)
  - wtraa(13,:)* volume of water leaving wetlands in HRU during simulation (mm H2O)
  - wtraa(14,:)* sediment loading from wetlands in HRU to main channel during simulation (metric tons/ha)
  - wtraa(15,:)* precipitation on potholes in HRU during simulation (mm H2O)
  - wtraa(16,:)* evaporation from potholes in HRU during simulation (mm H2O)
  - wtraa(17,:)* seepage from potholes in HRU during simulation (mm H2O)
  - wtraa(18,:)* water leaving potholes in HRU during simulation (mm H2O)
  - wtraa(19,:)* water entering potholes in HRU during simulation (mm H2O)
  - wtraa(20,:)* sediment entering potholes in HRU during simulation (metric tons/ha)
  - wtraa(21,:)* sediment leaving potholes in HRU during simulation (metric tons/ha)
- real \*8, dimension(:,:), allocatable [parm::sub\\_smfmx](#)
  - max melt rate for snow during year (June 21) for subbasin(:) where deg C refers to the air temperature. SUB\_SMFMX and SMFMN allow the rate of snow melt to vary through the year. These parameters are accounting for the impact of soil temperature on snow melt (range: -5.0/5.0) (mm/deg C/day)*
- real \*8, dimension(:,:), allocatable [parm::sub\\_smfmn](#)
  - min melt rate for snow during year (Dec 21) for subbasin(:) (range: -5.0/5.0) where deg C refers to the air temperature (mm/deg C/day)*
- real \*8, dimension(:,:), allocatable [parm::hrupstd](#)
  - HRU daily pesticide output array (varies)*
  - hrupstd(1,:,:) amount of pesticide type in surface runoff contribution to stream from HRU on day (in solution) (mg pst)*
  - hrupstd(2,:,:) amount of pesticide type in surface runoff contribution to stream from HRU on day (sorbed to sediment) (mg pst)*
  - hrupstd(3,:,:) total pesticide loading to stream in surface runoff from HRU (mg pst/ha)*
  - hrupstd(4,:,:) amount of pesticide type in lateral flow contribution to stream from HRU on day (in solution) (mg pst)*
- real \*8, dimension(:,:), allocatable [parm::hrupstm](#)
  - hrupstm(:,:,:) HRU monthly pesticide output array (varies)*
  - hrupstm(1,:,:) amount of pesticide type in surface runoff contribution to stream from HRU during month (in solution) (mg pst)*
  - hrupstm(2,:,:) amount of pesticide type in surface runoff contribution to stream from HRU during month (sorbed to sediment) (mg pst)*
  - hrupstm(3,:,:) total pesticide loading to stream in surface runoff from HRU during month (mg pst)*
- real \*8, dimension(:,:), allocatable [parm::hrupsta](#)
  - HRU average annual pesticide output array (varies)*
- real \*8, dimension(:,:), allocatable [parm::hrupsty](#)
  - hrupsty(:,:,:) HRU annual pesticide output array (varies)*
  - hrupsty(1,:,:) amount of pesticide type in surface runoff contribution to stream from HRU during year (in solution) (mg pst)*
  - hrupsty(2,:,:) amount of pesticide type in surface runoff contribution to stream from HRU during year (sorbed to sediment) (mg pst)*
- integer, dimension(:), allocatable [parm::ifirstt](#)
  - temperature data search code (none)*
  - 0 first day of temperature data located in file*
  - 1 first day of temperature data not located in file*

- integer, dimension(:), allocatable **parm::ifirstpcp**
- integer, dimension(:), allocatable **parm::elevp**  
*elevation of precipitation gage station (m)*
- integer, dimension(:), allocatable **parm::elevt**  
*elevation of temperature gage station (m)*
- real \*8, dimension(:,:), allocatable **parm::ftmpmn**  
*avg monthly minimum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **parm::ftmpmx**  
*avg monthly maximum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **parm::ftmpstdmn**  
*standard deviation for avg monthly minimum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **parm::ftmpstdmx**  
*standard deviation for avg monthly maximum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **parm::fpcp\_stat**  
*fpcp\_stat(:,1,:): average amount of precipitation falling in one day for the month (mm/day)*  
*fpcp\_stat(:,2,:): standard deviation for the average daily precipitation (mm/day)*  
*fpcp\_stat(:,3,:): skew coefficient for the average daily precipitation (none)*
- real \*8, dimension(:,:), allocatable **parm::fpr\_w**  
*fpr\_w(1,:,:) probability of wet day after dry day in month (none)*  
*fpr\_w(2,:,:) probability of wet day after wet day in month (none)*
- real \*8, dimension(:), allocatable **parm::ch\_d**  
*average depth of main channel (m)*
- real \*8, dimension(:), allocatable **parm::flwin**  
*flow into reach on current day ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable **parm::flwout**  
*flow out of reach on current day ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable **parm::bankst**  
*bank storage ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable **parm::ch\_wi**
- real \*8, dimension(:), allocatable **parm::ch\_onco**  
*channel organic n concentration (ppm)*
- real \*8, dimension(:), allocatable **parm::ch\_opco**  
*channel organic p concentration (ppm)*
- real \*8, dimension(:), allocatable **parm::ch\_orgn**
- real \*8, dimension(:), allocatable **parm::ch\_orgp**
- real \*8, dimension(:), allocatable **parm::rch\_dox**  
*dissolved oxygen concentration in reach (mg  $O_2/L$ )*
- real \*8, dimension(:), allocatable **parm::rch\_bactp**  
*persistent bacteria in reach/outflow at end of day (# cfu/100ml)*
- real \*8, dimension(:), allocatable **parm::alpha\_bnk**  
*alpha factor for bank storage recession curve (days)*
- real \*8, dimension(:), allocatable **parm::alpha\_bnke**  
 *$\exp(-\alpha_{bnk})$  (none)*
- real \*8, dimension(:), allocatable **parm::rchstor**  
*water stored in reach ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable **parm::sedst**  
*amount of sediment stored in reach (metric tons)*
- real \*8, dimension(:), allocatable **parm::algae**  
*algal biomass concentration in reach (mg alg/L)*
- real \*8, dimension(:), allocatable **parm::disolvp**  
*dissolved phosphorus concentration in reach (mg P/L)*
- real \*8, dimension(:), allocatable **parm::chlora**

- chlorophyll-a concentration in reach (mg chl-a/L)*
- real \*8, dimension(:), allocatable [parm::organic](#)
  - organic nitrogen concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [parm::organicp](#)
  - organic phosphorus concentration in reach (mg P/L)*
- real \*8, dimension(:), allocatable [parm::ch\\_li](#)
  - initial length of main channel (km)*
- real \*8, dimension(:), allocatable [parm::ch\\_si](#)
  - initial slope of main channel (m/m)*
- real \*8, dimension(:), allocatable [parm::nitraten](#)
  - nitrate concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [parm::nitriten](#)
  - nitrite concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable [parm::ch\\_bnk\\_san](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bnk\\_sil](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bnk\\_cla](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bnk\\_gra](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bed\\_san](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bed\\_sil](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bed\\_cla](#)
- real \*8, dimension(:), allocatable [parm::ch\\_bed\\_gra](#)
- real \*8, dimension(:), allocatable [parm::depfp](#)
- real \*8, dimension(:), allocatable [parm::depsilfp](#)
- real \*8, dimension(:), allocatable [parm::depclafp](#)
- real \*8, dimension(:), allocatable [parm::depch](#)
- real \*8, dimension(:), allocatable [parm::depsanch](#)
- real \*8, dimension(:), allocatable [parm::depsilch](#)
- real \*8, dimension(:), allocatable [parm::depclach](#)
- real \*8, dimension(:), allocatable [parm::depsagch](#)
- real \*8, dimension(:), allocatable [parm::deplagch](#)
- real \*8, dimension(:), allocatable [parm::depgrach](#)
- real \*8, dimension(:), allocatable [parm::grast](#)
- real \*8, dimension(:), allocatable [parm::prf](#)
  - Reach peak rate adjustment factor for sediment routing in the channel. Allows impact of peak flow rate on sediment routing and channel reshaping to be taken into account (none)*
- real \*8, dimension(:), allocatable [parm::depprch](#)
- real \*8, dimension(:), allocatable [parm::depprpf](#)
- real \*8, dimension(:), allocatable [parm::spcon](#)
  - linear parameter for calculating sediment reentrained in channel sediment routing*
- real \*8, dimension(:), allocatable [parm::spexp](#)
  - exponent parameter for calculating sediment reentrained in channel sediment routing*
- real \*8, dimension(:), allocatable [parm::sanst](#)
- real \*8, dimension(:), allocatable [parm::silst](#)
- real \*8, dimension(:), allocatable [parm::clast](#)
- real \*8, dimension(:), allocatable [parm::sagst](#)
- real \*8, dimension(:), allocatable [parm::lagst](#)
- real \*8, dimension(:), allocatable [parm::pot\\_san](#)
- real \*8, dimension(:), allocatable [parm::pot\\_sil](#)
- real \*8, dimension(:), allocatable [parm::pot\\_cla](#)
- real \*8, dimension(:), allocatable [parm::pot\\_sag](#)
- real \*8, dimension(:), allocatable [parm::pot\\_lag](#)
- real \*8, dimension(:), allocatable [parm::sanyld](#)
- real \*8, dimension(:), allocatable [parm::silyld](#)



- real \*8, dimension(:), allocatable **parm::clayld**
- real \*8, dimension(:), allocatable **parm::sagyld**
- real \*8, dimension(:), allocatable **parm::lagyld**
- real \*8, dimension(:), allocatable **parm::res\_san**
- real \*8, dimension(:), allocatable **parm::res\_sil**
- real \*8, dimension(:), allocatable **parm::res\_cla**
- real \*8, dimension(:), allocatable **parm::res\_sag**
- real \*8, dimension(:), allocatable **parm::res\_lag**
- real \*8, dimension(:), allocatable **parm::res\_gra**
- real \*8, dimension(:), allocatable **parm::pnd\_san**
- real \*8, dimension(:), allocatable **parm::pnd\_sil**
- real \*8, dimension(:), allocatable **parm::pnd\_cla**
- real \*8, dimension(:), allocatable **parm::pnd\_sag**
- real \*8, dimension(:), allocatable **parm::pnd\_lag**
- real \*8, dimension(:), allocatable **parm::wet\_san**
- real \*8, dimension(:), allocatable **parm::wet\_sil**
- real \*8, dimension(:), allocatable **parm::wet\_cla**
- real \*8, dimension(:), allocatable **parm::wet\_lag**
- real \*8, dimension(:), allocatable **parm::wet\_sag**
- real \*8 **parm::ressani**
- real \*8 **parm::ressili**
- real \*8 **parm::resclai**
- real \*8 **parm::ressagi**
- real \*8 **parm::reslagi**
- real \*8 **parm::resgrai**
- real \*8 **parm::pndsarin**
- real \*8 **parm::pndsilin**
- real \*8 **parm::pndclain**
- real \*8 **parm::pndsagin**
- real \*8 **parm::pndlagin**
- real \*8 **parm::pndsano**
- real \*8 **parm::pndsilo**
- real \*8 **parm::pndclao**
- real \*8 **parm::pndsago**
- real \*8 **parm::pndlago**
- real \*8, dimension(:), allocatable **parm::ch\_di**  
*initial depth of main channel (m)*
- real \*8, dimension(:, :), allocatable **parm::ch\_l**  
*ch\_l(1,:) longest tributary channel length in subbasin (km)*  
*ch\_l(2,:) length of main channel (km)*
- real \*8, dimension(:), allocatable **parm::ch\_bnk\_bd**  
*bulk density of channel bank sediment (1.1-1.9) (g/cc)*
- real \*8, dimension(:), allocatable **parm::ch\_bed\_bd**  
*bulk density of channel bed sediment (1.1-1.9) (g/cc)*
- real \*8, dimension(:), allocatable **parm::ch\_bnk\_kd**  
*erodibility of channel bank sediment by jet test (Peter Allen needs to give more info on this)*
- real \*8, dimension(:), allocatable **parm::ch\_bed\_kd**  
*erodibility of channel bed sediment by jet test (Peter Allen needs to give more info on this)*
- real \*8, dimension(:), allocatable **parm::ch\_bnk\_d50**  
*D50(median) particle size diameter of channel bank sediment (0.001 - 20)*
- real \*8, dimension(:), allocatable **parm::ch\_bed\_d50**  
*D50(median) particle size diameter of channel bed sediment (micrometers) (0.001 - 20)*
- real \*8, dimension(:, :), allocatable **parm::ch\_cov**

- ch\_cov(1,:) channel erodibility factor (0.0-1.0) (none)*
  - 0 non-erosive channel*
  - 1 no resistance to erosion*
- ch\_cov(2,:) channel cover factor (0.0-1.0) (none)*
  - 0 channel is completely protected from erosion by cover*
  - 1 no vegetative cover on channel*
- real \*8, dimension(:), allocatable [parm::tc\\_bed](#)
  - critical shear stress of channel bed (N/m2)*
- real \*8, dimension(:), allocatable [parm::tc\\_bnk](#)
  - critical shear stress of channel bank (N/m2)*
- integer, dimension(:), allocatable [parm::ch\\_eqn](#)
  - sediment routine methods (DAILY):*
    - 0 = original SWAT method*
    - 1 = Bagnold's*
    - 2 = Kodatie*
    - 3 = Molinas WU*
    - 4 = Yang*
- real \*8, dimension(:), allocatable [parm::chpst\\_rea](#)
  - pesticide reaction coefficient in reach (1/day)*
- real \*8, dimension(:), allocatable [parm::chpst\\_vol](#)
  - pesticide volatilization coefficient in reach (m/day)*
- real \*8, dimension(:), allocatable [parm::chpst\\_conc](#)
  - initial pesticide concentration in reach (mg/(m<sup>3</sup>))*
- real \*8, dimension(:), allocatable [parm::chpst\\_koc](#)
  - pesticide partition coefficient between water and sediment in reach (m<sup>3</sup>/g)*
- real \*8, dimension(:), allocatable [parm::chpst\\_rsp](#)
  - resuspension velocity in reach for pesticide sorbed to sediment (m/day)*
- real \*8, dimension(:), allocatable [parm::chpst\\_stl](#)
  - settling velocity in reach for pesticide sorbed to sediment (m/day)*
- real \*8, dimension(:), allocatable [parm::ch\\_wdr](#)
  - channel width to depth ratio (m/m)*
- real \*8, dimension(:), allocatable [parm::chpst\\_mix](#)
  - mixing velocity (diffusion/dispersion) for pesticide in reach (m/day)*
- real \*8, dimension(:), allocatable [parm::sedpst\\_conc](#)
  - initial pesticide concentration in river bed sediment (mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable [parm::sedpst\\_bry](#)
  - pesticide burial velocity in river bed sediment (m/day)*
- real \*8, dimension(:), allocatable [parm::sedpst\\_rea](#)
  - pesticide reaction coefficient in river bed sediment (1/day)*
- real \*8, dimension(:), allocatable [parm::sedpst\\_act](#)
  - depth of active sediment layer in reach for pesticide (m)*
- real \*8, dimension(:), allocatable [parm::rch\\_cbod](#)
  - carbonaceous biochemical oxygen demand in reach (mg O2/L)*
- real \*8, dimension(:), allocatable [parm::rch\\_bactlp](#)
  - less persistent bacteria in reach/outflow at end of day (# cfu/100ml)*
- real \*8, dimension(:), allocatable [parm::chside](#)
  - change in horizontal distance per unit vertical distance (0.0 - 5)*
    - 0 = for vertical channel bank*
    - 5 = for channel bank with gentl side slope*
- real \*8, dimension(:,:), allocatable [parm::rs](#)

- rs(1,:) local algal settling rate in reach at 20 deg C (m/day or m/hour)*
- rs(2,:) benthos source rate for dissolved phosphorus in reach at 20 deg C ((mg disP-P)/(m<sup>2</sup>\*day) or (mg disP-P)/(m<sup>2</sup>\*hour))*
- rs(3,:) benthos source rate for ammonia nitrogen in reach at 20 deg C ((mg NH4-N)/(m<sup>2</sup>\*day) or (mg NH4-N)/(m<sup>2</sup>\*hour))*
- rs(4,:) rate coefficient for organic nitrogen settling in reach at 20 deg C (1/day or 1/hour)*
- rs(5,:) organic phosphorus settling rate in reach at 20 deg C (1/day or 1/hour)*
- rs(6,:) rate coefficient for settling of arbitrary non-conservative constituent in reach (1/day)*
- rs(7,:) benthos source rate for arbitrary non-conservative constituent in reach ((mg ANC)/(m<sup>2</sup>\*day))*
- real \*8, dimension(:,:), allocatable **parm::rk**
  - rk(1,:) CBOD deoxygenation rate coefficient in reach at 20 deg C (1/day or 1/hour)*
  - rk(2,:) reaeration rate in accordance with Fickian diffusion in reach at 20 deg C (1/day or 1/hour)*
  - rk(3,:) rate of loss of CBOD due to settling in reach at 20 deg C (1/day or 1/hour)*
  - rk(4,:) sediment oxygen demand rate in reach at 20 deg C (mg O2/(m<sup>2</sup>\*day) or mg O2/(m<sup>2</sup>\*hour))*
  - rk(5,:) coliform die-off rate in reach (1/day)*
  - rk(6,:) decay rate for arbitrary non-conservative constituent in reach (1/day)*
- real \*8, dimension(:,:), allocatable **parm::bc**
  - bc(1,:) rate constant for biological oxidation of NH3 to NO2 in reach at 20 deg C (1/day or 1/hour)*
  - bc(2,:) rate constant for biological oxidation of NO2 to NO3 in reach at 20 deg C (1/day or 1/hour)*
  - bc(3,:) rate constant for hydrolysis of organic N to ammonia in reach at 20 deg C (1/day or 1/hour)*
  - bc(4,:) rate constant for the decay of organic P to dissolved P in reach at 20 deg C (1/day or 1/hour)*
- real \*8, dimension(:), allocatable **parm::ammonian**
  - ammonia concentration in reach (mg N/L)*
- real \*8, dimension(:), allocatable **parm::orig\_sedpstconc**
- real \*8, dimension(:,:), allocatable **parm::wurch**
  - average daily water removal from the reach for the month (10<sup>4</sup> m<sup>3</sup>/day)*
- integer, dimension(:), allocatable **parm::icanal**
- integer, dimension(:), allocatable **parm::itb**
- real \*8, dimension(:), allocatable **parm::ch\_revap**
  - revap coeff: this variable controls the amount of water moving from bank storage to the root zone as a result of soil moisture depletion (none)*
- real \*8, dimension(:), allocatable **parm::dep\_chan**
- real \*8, dimension(:), allocatable **parm::harg\_petco**
  - coefficient related to radiation used in hargreaves eq (range: 0.0019 - 0.0032)*
- real \*8, dimension(:), allocatable **parm::subfr\_nowtr**
- real \*8, dimension(:), allocatable **parm::cncoef\_sub**
  - soil water depletion coefficient used in the new (modified curve number method) same as soil index coeff used in APEX range: 0.5 - 2.0*
- real \*8, dimension(:), allocatable **parm::dr\_sub**
- real \*8, dimension(:), allocatable **parm::sub\_fr**
  - fraction of total watershed area contained in subbasin (km<sup>2</sup>/km<sup>2</sup>)*
- real \*8, dimension(:), allocatable **parm::sub\_sw**
  - amount of water in soil profile on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable **parm::wcklsp**
- real \*8, dimension(:), allocatable **parm::sub\_gwno3**
  - nitrate loading in groundwater from subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable **parm::sub\_sumfc**
  - amount of water in soil at field capacity in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable **parm::sub\_gwsolp**
- real \*8, dimension(:), allocatable **parm::co2**
  - CO2 concentration (ppmv)*
- real \*8, dimension(:), allocatable **parm::sub\_km**
  - area of subbasin in square kilometers (km<sup>2</sup>)*
- real \*8, dimension(:), allocatable **parm::sub\_tc**
  - time of concentration for subbasin (hour)*

- real \*8, dimension(:), allocatable [parm::sub\\_pet](#)  
*potential evapotranspiration for day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::welev](#)  
*elevation of weather station used to compile weather generator data (m)*
- real \*8, dimension(:), allocatable [parm::sub\\_bd](#)  
*average bulk density in subbasin for top 10 mm of first soil layer ( $\text{Mg/m}^3$ )*
- real \*8, dimension(:), allocatable [parm::sub\\_orgn](#)  
*amount of nitrogen stored in all organic pools in soil of subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_orgp](#)  
*amount of phosphorus stored in all organic pools in soil of subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_sedpa](#)  
*amount of active mineral P attached to sediment removed in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_sedps](#)  
*amount of stable mineral P attached to sediment removed in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable **parm::sub\_wtmp**
- real \*8, dimension(:), allocatable [parm::daylmn](#)  
*shortest daylength occurring during the year (hour)*
- real \*8, dimension(:), allocatable [parm::sub\\_minpa](#)  
*amount of phosphorus stored in active mineral pools sorbed to sediment (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_minps](#)  
*amount of phosphorus stored in stable mineral pools sorbed to sediment (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::latcos](#)  
 *$\cos(\text{latitude})$  (none)*
- real \*8, dimension(:), allocatable [parm::latsin](#)  
 *$\sin(\text{latitude})$  (none)*
- real \*8, dimension(:), allocatable [parm::phutot](#)  
*total potential heat units for year (used when no crop is growing) (heat unit)*
- real \*8, dimension(:), allocatable [parm::plaps](#)  
*precipitation lapse rate: precipitation change due to change in elevation (mm H2O/km)*
- real \*8, dimension(:), allocatable [parm::tlaps](#)  
*temperature lapse rate: temperature change due to change in elevation (deg C/km)*
- real \*8, dimension(:), allocatable [parm::tmp\\_an](#)  
*average annual air temperature (deg C)*
- real \*8, dimension(:), allocatable [parm::sub\\_precip](#)  
*effective precipitation (amount of water reaching soil surface) for the day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::rammo\\_sub](#)  
*atmospheric deposition of ammonium values for entire watershed (mg/l)*
- real \*8, dimension(:), allocatable [parm::rcn\\_sub](#)  
*atmospheric deposition of nitrate for entire watershed (mg/l)*
- real \*8, dimension(:), allocatable **parm::pcpdays**
- real \*8, dimension(:), allocatable [parm::sub\\_snom](#)  
*amount of snow melt in subbasin on day (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_qd](#)  
*surface runoff that reaches main channel during day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_sedy](#)  
*sediment yield for the day in subbasin (metric tons)*
- real \*8, dimension(:), allocatable [parm::sub\\_tran](#)  
*transmission losses on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_no3](#)  
 *$\text{NO}_3\text{-N}$  in surface runoff on day in subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_latno3](#)

- NO3-N in lateral flow on day in subbasin (kg N/ha)*
- real \*8, dimension(:,:), allocatable [parm::sub\\_sftmp](#)  
*snowfall temperature for subbasin(:). Mean air temperature at which precip is equally likely to be rain as snow/freezing rain (range: -5.0/5.0) (deg C)*
- real \*8, dimension(:,:), allocatable [parm::sub\\_smtmp](#)  
*snow melt base temperature for subbasin(:) mean air temperature at which snow melt will occur (range: -5.0/5.0) (deg C)*
- real \*8, dimension(:,:), allocatable [parm::sub\\_timp](#)  
*snow pack temperature lag factor (0-1) (none)*  
*1 = no lag (snow pack temp=current day air temp) as the lag factor goes to zero, the snow pack's temperature will be less influenced by the current day's air temperature*
- real \*8, dimension(:), allocatable [parm::sub\\_tileno3](#)  
*NO3 in tile flow on day in subbasin (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_etday](#)  
*actual evapotranspiration on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_solp](#)  
*soluble P in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_subp](#)  
*precipitation for day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_elev](#)  
*average elevation of HRU (m)*
- real \*8, dimension(:), allocatable [parm::sub\\_surfq](#)  
*surface runoff generated on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_wyld](#)  
*water yield on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable **parm::qird**
- real \*8, dimension(:), allocatable [parm::sub\\_gwq](#)  
*groundwater flow on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_sep](#)  
*seepage from bottom of soil profile on day in subbasin (mm H2O)*
- real \*8, dimension(:), allocatable [parm::sub\\_chl](#)  
*chlorophyll-a in water yield on day in subbasin (kg chl-a)*
- real \*8, dimension(:), allocatable [parm::sub\\_cbod](#)  
*carbonaceous biological oxygen demand loading on day for subbasin (kg O2)*
- real \*8, dimension(:), allocatable [parm::sub\\_dox](#)  
*dissolved oxygen loading on day for subbasin (kg O2)*
- real \*8, dimension(:), allocatable [parm::sub\\_solpst](#)  
*pesticide in solution in surface runoff on day in subbasin (mg pst)*
- real \*8, dimension(:), allocatable [parm::sub\\_yorgn](#)  
*organic N loading in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_yorgp](#)  
*organic P loading in surface runoff on day in subbasin (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sub\\_sorpst](#)  
*pesticide sorbed to sediment in surface runoff on day in subbasin (mg pst)*
- real \*8, dimension(:), allocatable [parm::sub\\_lat](#)  
*latitude of HRU/subbasin (degrees)*
- real \*8, dimension(:), allocatable [parm::sub\\_bactlp](#)  
*less persistent bacteria in surface runoff for day in subbasin (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable [parm::sub\\_bactp](#)  
*persistent bacteria in surface runoff for day in subbasin (# cfu/m<sup>2</sup>)*
- real \*8, dimension(:), allocatable **parm::sub\_latq**
- real \*8, dimension(:), allocatable **parm::sub\_gwq\_d**

- real \*8, dimension(:), allocatable **parm::sub\_tileq**
- real \*8, dimension(:), allocatable **parm::sub\_vaptile**
- real \*8, dimension(:), allocatable **parm::sub\_dsan**
- real \*8, dimension(:), allocatable **parm::sub\_dsil**
- real \*8, dimension(:), allocatable **parm::sub\_dcla**
- real \*8, dimension(:), allocatable **parm::sub\_dsag**
- real \*8, dimension(:), allocatable **parm::sub\_dlag**
- real \*8 **parm::vap\_tile**
- real \*8, dimension(:,:), allocatable **parm::sol\_stpwt**
- real \*8, dimension(:,:), allocatable **parm::sub\_hhwtmp**  
*water temperature for the time step in subbasin (deg C)*
- real \*8, dimension(:,:), allocatable **parm::sub\_hhq**
- real \*8, dimension(:,:), allocatable **parm::huminc**  
*monthly humidity adjustment. Daily values for relative humidity within the month are raised or lowered by the specified amount (used in climate change studies) (none)*
- real \*8, dimension(:,:), allocatable **parm::radinc**  
*monthly solar radiation adjustment. Daily radiation within the month is raised or lowered by the specified amount (used in climate change studies) (MJ/m<sup>2</sup>)*
- real \*8, dimension(:,:), allocatable **parm::rfinc**  
*monthly rainfall adjustment. Daily rainfall within the month is adjusted to the specified percentage of the original value (used in climate change studies)(%)*
- real \*8, dimension(:,:), allocatable **parm::tmpinc**  
*monthly temperature adjustment. Daily maximum and minimum temperatures within the month are raised or lowered by the specified amount (used in climate change studies) (deg C)*
- real \*8, dimension(:,:), allocatable **parm::ch\_k**  
*ch\_k(1,:) effective hydraulic conductivity of tributary channel alluvium (mm/hr)*  
*ch\_k(2,:) effective hydraulic conductivity of main channel alluvium (mm/hr)*
- real \*8, dimension(:,:), allocatable **parm::elevb**  
*elevation at the center of the band in subbasin (m)*
- real \*8, dimension(:,:), allocatable **parm::elevb\_fr**  
*fraction of subbasin area within elevation band (the same fractions should be listed for all HRUs within the subbasin) (none)*
- real \*8, dimension(:,:), allocatable **parm::wndav**  
*average wind speed for the month (m/s)*
- real \*8, dimension(:,:), allocatable **parm::ch\_n**  
*ch\_n(1,:) Manning's "n" value for the tributary channels (none)*  
*ch\_n(2,:) Manning's "n" value for the main channel (none)*
- real \*8, dimension(:,:), allocatable **parm::ch\_s**  
*ch\_s(1,:) average slope of tributary channels (m/m)*  
*ch\_s(2,:) average slope of main channel (m/m)*
- real \*8, dimension(:,:), allocatable **parm::ch\_w**  
*ch\_w(1,:) average width of tributary channels (m)*  
*ch\_w(2,:) average width of main channel (m)*
- real \*8, dimension(:,:), allocatable **parm::dewpt**  
*average dew point temperature for the month (deg C)*
- real \*8, dimension(:,:), allocatable **parm::amp\_r**  
*average fraction of total daily rainfall occurring in maximum half-hour period for month (none)*
- real \*8, dimension(:,:), allocatable **parm::solarav**  
*average daily solar radiation for the month (MJ/m<sup>2</sup>/day)*
- real \*8, dimension(:,:), allocatable **parm::tmpstdmx**  
*standard deviation for avg monthly maximum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable **parm::pcf**  
*normalization coefficient for precipitation generated from skewed distribution (none)*

- real \*8, dimension(:,:), allocatable [parm::tmpmn](#)  
*avg monthly minimum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable [parm::tmpmx](#)  
*avg monthly maximum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable [parm::tmpstdmn](#)  
*standard deviation for avg monthly minimum air temperature (deg C)*
- real \*8, dimension(:,:), allocatable [parm::otmpstdmn](#)
- real \*8, dimension(:,:), allocatable [parm::otmpmn](#)
- real \*8, dimension(:,:), allocatable [parm::otmpmx](#)
- real \*8, dimension(:,:), allocatable [parm::otmpstdmx](#)
- real \*8, dimension(:,:), allocatable [parm::ch\\_erodmo](#)
- real \*8, dimension(:,:), allocatable [parm::uh](#)
- real \*8, dimension(:,:), allocatable [parm::hqdsave](#)
- real \*8, dimension(:,:), allocatable [parm::hsdsave](#)
- real \*8, dimension(:,:), allocatable [parm::pr\\_w](#)  
*pr\_w(1,:,:) probability of wet day after dry day in month (none)*  
*pr\_w(2,:,:) probability of wet day after wet day in month (none)*  
*pr\_w(3,:,:) proportion of wet days in the month (none)*
- real \*8, dimension(:,:), allocatable [parm::pcp\\_stat](#)
- real \*8, dimension(:,:), allocatable [parm::opr\\_w](#)
- real \*8, dimension(:,:), allocatable [parm::opcp\\_stat](#)
- integer, dimension(:), allocatable [parm::ireg](#)  
*precipitation category (none):*  
*1 precipitation <= 508 mm/yr*  
*2 precipitation > 508 and <= 1016 mm/yr*  
*3 precipitation > 1016 mm/yr*
- integer, dimension(:), allocatable [parm::hrutot](#)  
*number of HRUs in subbasin (none)*
- integer, dimension(:), allocatable [parm::hru1](#)
- integer, dimension(:), allocatable [parm::ihgage](#)  
*HRU relative humidity data code (gage # for relative humidity data used in as HRU) (none)*
- integer, dimension(:), allocatable [parm::isgage](#)  
*HRU solar radiation data code (record # for solar radiation used in HRU) (none)*
- integer, dimension(:), allocatable [parm::iwgage](#)  
*HRU wind speed gage data code (gage # for wind speed data used in HRU) (none)*
- integer, dimension(:), allocatable [parm::subgis](#)  
*GIS code printed to output files (output.sub, .rch) (none)*
- integer, dimension(:), allocatable [parm::irgage](#)  
*subbasin rain gage data code (gage # for rainfall data used in HRU) (none)*
- integer, dimension(:), allocatable [parm::itgage](#)  
*subbasin temp gage data code (gage # for temperature data used in HRU) (none)*
- integer, dimension(:), allocatable [parm::irelh](#)  
*(none) irelh = 0 (dewpoint)*  
*irelh = 1 (relative humidity)*  
*note: inputs > 1.0 (dewpoint)*  
*inputs < 1.0 (relative hum)*
- integer, dimension(:), allocatable [parm::fcst\\_reg](#)
- real \*8, dimension(:,:), allocatable [parm::sol\\_aorgn](#)  
*amount of nitrogen stored in the active organic (humic) nitrogen pool in soil layer (kg N/ha)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_fon](#)  
*amount of nitrogen stored in the fresh organic (residue) pool in soil layer (kg N/ha)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_tmp](#)  
*average temperature of soil layer on previous day or*  
*daily average temperature of soil layer (deg C)*



- real \*8, dimension(:,:), allocatable [parm::sol\\_awc](#)  
*available water capacity of soil layer (mm H2O/mm soil)*
- real \*8, dimension(:,:), allocatable [parm::volcr](#)  
*crack volume for soil layer (mm)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_prk](#)  
*percolation storage from soil layer on current day (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::pperco\\_sub](#)  
*subbasin phosphorus percolation coefficient. Ratio of soluble phosphorus in surface to soluble phosphorus in percolate*
- real \*8, dimension(:,:), allocatable [parm::sol\\_stap](#)  
*amount of phosphorus in the soil layer stored in the stable mineral phosphorus pool (kg P/ha)*
- real \*8, dimension(:,:), allocatable [parm::conv\\_wt](#)  
*factor which converts kg/kg soil to kg/ha (none)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_actp](#)  
*amount of phosphorus stored in the active mineral phosphorus pool (kg P/ha)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_solp](#)  
*soluble P concentration in top soil layer (mg P/kg soil) or  
amount of inorganic phosphorus stored in solution in soil layer. NOTE UNIT CHANGE! (kg P/ha)*
- real \*8, dimension(:,:), allocatable [parm::crdep](#)  
*maximum or potential crack volume (mm)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_fc](#)  
*amount of water available to plants in soil layer at field capacity (fc - wp water) (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_ul](#)  
*amount of water held in the soil layer at saturation (sat - wp water) (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_bd](#)  
*bulk density of the soil layer in HRU ( $\text{Mg/m}^3$ )*
- real \*8, dimension(:,:), allocatable [parm::sol\\_z](#)  
*depth to bottom of each soil profile layer in a given HRU (mm)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_st](#)  
*amount of water stored in the soil layer on any given day (less wilting point water) (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_up](#)  
*water content of soil at -0.033 MPa (field capacity) (mm H2O/mm soil)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_clay](#)  
*percent clay content in soil layer in HRU (UNIT CHANGE!) (% or none)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_hk](#)  
*beta coefficient to calculate hydraulic conductivity (none)*
- real \*8, dimension(:,:), allocatable [parm::flat](#)  
*lateral flow storage in soil layer on current day (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_nh3](#)  
*amount of nitrogen stored in the ammonium pool in soil layer (kg N/ha)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_ec](#)  
*electrical conductivity of soil layer (dS/m)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_orn](#)  
*amount of nitrogen stored in the stable organic N pool. NOTE UNIT CHANGE! (mg N/kg soil or kg N/ha)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_por](#)  
*total porosity of soil layer expressed as a fraction of the total volume (none)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_wp](#)  
*water content of soil at -1.5 MPa (wilting point) (mm H2O/mm soil)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_organ](#)  
*amount of phosphorus stored in the organic P pool in soil layer. NOTE UNIT CHANGE! (mg P/kg soil or kg P/ha)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_wpm](#)



- water content of soil at -1.5 MPa (wilting point) (mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::sol\_no3**  
*amount of nitrogen stored in the nitrate pool in the soil layer. This variable is read in as a concentration and converted to kg/ha (this value is read from the .sol file in units of mg/kg) (kg N/ha)*
- real \*8, dimension(:,:), allocatable **parm::sol\_cbn**  
*percent organic carbon in soil layer (%)*
- real \*8, dimension(:,:), allocatable **parm::sol\_k**  
*saturated hydraulic conductivity of soil layer (mm/hour)*
- real \*8, dimension(:,:), allocatable **parm::sol\_rsd**  
*amount of organic matter in the soil layer classified as residue (kg/ha)*
- real \*8, dimension(:,:), allocatable **parm::sol\_fop**  
*amount of phosphorus stored in the fresh organic (residue) pool in soil layer (kg P/ha)*
- real \*8, dimension(:,:), allocatable **parm::sol\_rock**  
*percent of rock fragments in soil layer (%)*
- real \*8, dimension(:,:), allocatable **parm::sol\_silt**  
*percent silt content in soil material (UNIT CHANGE!) (% or none)*
- real \*8, dimension(:,:), allocatable **parm::sol\_sand**  
*percent sand content of soil material (%)*
- real \*8, dimension(:,:), allocatable **parm::orig\_solno3**
- real \*8, dimension(:,:), allocatable **parm::orig\_solorgn**
- real \*8, dimension(:,:), allocatable **parm::orig\_solsofp**
- real \*8, dimension(:,:), allocatable **parm::orig\_solorgp**
- real \*8, dimension(:,:), allocatable **parm::orig\_soltmp**
- real \*8, dimension(:,:), allocatable **parm::orig\_solrsd**
- real \*8, dimension(:,:), allocatable **parm::orig\_solfop**
- real \*8, dimension(:,:), allocatable **parm::orig\_solfon**
- real \*8, dimension(:,:), allocatable **parm::orig\_solaorgn**
- real \*8, dimension(:,:), allocatable **parm::orig\_solst**
- real \*8, dimension(:,:), allocatable **parm::orig\_solactp**
- real \*8, dimension(:,:), allocatable **parm::orig\_solstap**
- real \*8, dimension(:,:), allocatable **parm::orig\_volcr**
- real \*8, dimension(:,:), allocatable **parm::sol\_pst**  
*sol\_pst(:,1) initial amount of pesticide in first layer read in from .chm file (mg/kg)*  
*sol\_pst(:,2) amount of pesticide in soil layer. NOTE UNIT CHANGE! (kg/ha)*
- real \*8, dimension(:,:), allocatable **parm::sol\_kp**  
*pesticide sorption coefficient, Kp; the ratio of the concentration in the solid phase to the concentration in solution ((mg/kg)/(mg/L) or m<sup>3</sup>/ton)*
- real \*8, dimension(:,:), allocatable **parm::orig\_solpst**
- real \*8, dimension(:), allocatable **parm::velsetlr**
- real \*8, dimension(:), allocatable **parm::velsetlp**
- real \*8, dimension(:,:), allocatable **parm::br**  
*br(1,:) 1st shape parameter for reservoir surface area equation (none)*  
*br(2,:) 2nd shape parameter for reservoir surface area equation (none)*
- real \*8, dimension(:), allocatable **parm::evrsv**  
*lake evaporation coefficient (none)*
- real \*8, dimension(:), allocatable **parm::res\_k**  
*hydraulic conductivity of the reservoir bottom (mm/hr)*
- real \*8, dimension(:), allocatable **parm::lkpst\_conc**  
*pesticide concentration in lake water (mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **parm::res\_evol**  
*volume of water needed to fill the reservoir to the emergency spillway (read in as 10<sup>4</sup> m<sup>3</sup> and converted to m<sup>3</sup>) (m<sup>3</sup>)*
- real \*8, dimension(:), allocatable **parm::res\_pvol**

- volume of water needed to fill the reservoir to the principal spillway (read in as  $10^4 \text{ m}^3$  and converted to  $\text{m}^3$ ) ( $\text{m}^3$ )*
- real \*8, dimension(:), allocatable [parm::res\\_vol](#)  
*reservoir volume (read in as  $10^4 \text{ m}^3$  and converted to  $\text{m}^3$ ) ( $\text{m}^3$ )*
  - real \*8, dimension(:), allocatable [parm::res\\_psa](#)  
*reservoir surface area when reservoir is filled to principal spillway (ha)*
  - real \*8, dimension(:), allocatable [parm::lkpst\\_rea](#)  
*pesticide reaction coefficient in lake water (1/day)*
  - real \*8, dimension(:), allocatable [parm::lkpst\\_vol](#)  
*pesticide volatilization coefficient in lake water (m/day)*
  - real \*8, dimension(:), allocatable [parm::res\\_rr](#)  
*average daily principal spillway release volume (read in as a release rate in  $\text{m}^3/\text{s}$  and converted to  $\text{m}^3/\text{day}$ ) ( $\text{m}^3/\text{day}$ )*
  - real \*8, dimension(:), allocatable [parm::res\\_sed](#)  
*amount of sediment in reservoir (read in as mg/L and converted to kg/L) (kg/L)*
  - real \*8, dimension(:), allocatable [parm::lkpst\\_koc](#)  
*pesticide partition coefficient between water and sediment in lake water ( $\text{m}^3/\text{g}$ )*
  - real \*8, dimension(:), allocatable [parm::lkpst\\_mix](#)  
*mixing velocity (diffusion/dispersion) in lake water for pesticide (m/day)*
  - real \*8, dimension(:), allocatable [parm::lkpst\\_rsp](#)  
*resuspension velocity in lake water for pesticide sorbed to sediment (m/day)*
  - real \*8, dimension(:), allocatable [parm::lkpst\\_stl](#)  
*settling velocity in lake water for pesticide sorbed to sediment (m/day)*
  - real \*8, dimension(:), allocatable [parm::lkspst\\_conc](#)  
*pesticide concentration in lake bed sediment ( $\text{mg}/\text{m}^3$ )*
  - real \*8, dimension(:), allocatable [parm::lkspst\\_rea](#)  
*pesticide reaction coefficient in lake bed sediment (1/day)*
  - real \*8, dimension(:), allocatable [parm::theta\\_n](#)
  - real \*8, dimension(:), allocatable [parm::theta\\_p](#)
  - real \*8, dimension(:), allocatable [parm::con\\_nirr](#)
  - real \*8, dimension(:), allocatable [parm::con\\_pirr](#)
  - real \*8, dimension(:), allocatable [parm::lkspst\\_act](#)  
*depth of active sediment layer in lake for for pesticide (m)*
  - real \*8, dimension(:), allocatable [parm::lkspst\\_bry](#)  
*pesticide burial velocity in lake bed sediment (m/day)*
  - real \*8, dimension(:), allocatable [parm::sed\\_stlr](#)
  - real \*8, dimension(7) [parm::resdata](#)  
*resdata(1) average annual evaporation from reservoirs in watershed (mm H2O)*  
*resdata(2) average annual seepage from reservoirs in watershed (mm H2O)*  
*resdata(3) average annual precipitation on reservoirs in watershed (mm H2O)*  
*resdata(4) average annual amount of water transported into reservoirs in watershed (mm H2O)*  
*resdata(5) average annual amount of sediment transported into reservoirs in watershed (metric tons/ha)*  
*resdata(6) average annual amount of water transported out of reservoirs in watershed (mm H2O)*  
*resdata(7) average annual amount of sediment transported out of reservoirs in watershed (metric tons/ha)*
  - real \*8, dimension(:), allocatable [parm::res\\_nsed](#)  
*normal amount of sediment in reservoir (read in as mg/L and convert to kg/L) (kg/L)*
  - real \*8, dimension(:), allocatable [parm::wurtmf](#)  
*fraction of water removed from the reservoir via WURESN which is returned and becomes flow from the reservoir outlet (none)*
  - real \*8, dimension(:), allocatable [parm::chlar](#)  
*chlorophyll-a production coefficient for reservoir (none)*
  - real \*8, dimension(:), allocatable [parm::res\\_no3](#)  
*amount of nitrate in reservoir (kg N)*

- real \*8, dimension(:), allocatable [parm::res\\_orgn](#)  
*amount of organic N in reservoir (kg N)*
- real \*8, dimension(:), allocatable [parm::res\\_orgp](#)  
*amount of organic P in reservoir (kg P)*
- real \*8, dimension(:), allocatable [parm::res\\_solp](#)  
*amount of soluble P in reservoir (kg P)*
- real \*8, dimension(:), allocatable [parm::res\\_seci](#)  
*secchi-disk depth (m)*
- real \*8, dimension(:), allocatable [parm::res\\_nh3](#)  
*amount of ammonia in reservoir (kg N)*
- real \*8, dimension(:), allocatable [parm::res\\_no2](#)  
*amount of nitrite in reservoir (kg N)*
- real \*8, dimension(:), allocatable [parm::seccir](#)  
*water clarity coefficient for reservoir (none)*
- real \*8, dimension(:), allocatable [parm::oflowmn\\_fps](#)  
*minimum reservoir outflow as a fraction of the principal spillway volume (0-1) (fraction)*
- real \*8, dimension(:), allocatable [parm::starg\\_fps](#)  
*target volume as a fraction of the principal spillway volume (.1-5) (fraction)*
- real \*8, dimension(:), allocatable [parm::weirc](#)
- real \*8, dimension(:), allocatable [parm::weirk](#)
- real \*8, dimension(:), allocatable [parm::weirw](#)
- real \*8, dimension(:), allocatable [parm::acof](#)
- real \*8, dimension(:), allocatable [parm::bcof](#)
- real \*8, dimension(:), allocatable [parm::ccoef](#)
- real \*8, dimension(:), allocatable [parm::orig\\_resvol](#)
- real \*8, dimension(:), allocatable [parm::orig\\_ressed](#)
- real \*8, dimension(:), allocatable [parm::orig\\_lkpstconc](#)
- real \*8, dimension(:), allocatable [parm::orig\\_lkspstconc](#)
- real \*8, dimension(:), allocatable [parm::orig\\_ressolp](#)
- real \*8, dimension(:), allocatable [parm::orig\\_resorgp](#)
- real \*8, dimension(:), allocatable [parm::orig\\_resno3](#)
- real \*8, dimension(:), allocatable [parm::orig\\_resno2](#)
- real \*8, dimension(:), allocatable [parm::orig\\_resnh3](#)
- real \*8, dimension(:), allocatable [parm::orig\\_resorgn](#)
- real \*8, dimension(:,), allocatable [parm::oflowmn](#)  
*minimum daily outflow for the month (read in as  $m^3/s$  and converted to  $m^3/day$ ) ( $m^3/day$ )*
- real \*8, dimension(:,), allocatable [parm::oflowmx](#)  
*maximum daily outflow for the month (read in as  $m^3/s$  and converted to  $m^3/day$ ) ( $m^3/day$ )*
- real \*8, dimension(:,), allocatable [parm::starg](#)  
*monthly target reservoir storage (needed if IRESCO=2) (read in as  $10^4 m^3$  and converted to  $m^3$ ) ( $m^3$ )*
- real \*8, dimension(:,), allocatable [parm::psetlr](#)  
*psetlr(1,:) phosphorus settling rate for mid-year period (read in as m/year and converted to m/day) (m/day)*  
*psetlr(2,:) phosphorus settling rate for remainder of year (read in as m/year and converted to m/day) (m/day)*
- real \*8, dimension(:,), allocatable [parm::nsetlr](#)  
*nsetlr(1,:) nitrogen settling rate for mid-year period (read in as m/year and converted to m/day) (m/day)*  
*nsetlr(2,:) nitrogen settling rate for remainder of year (read in as m/year and converted to m/day) (m/day)*
- real \*8, dimension(:,), allocatable [parm::wuresn](#)  
*average amount of water withdrawn from reservoir each month for consumptive water use (read in as  $10^4 m^3$  and converted to  $m^3$ ) ( $m^3$ )*
- real \*8, dimension(:,,:), allocatable [parm::res\\_out](#)  
*measured average daily outflow from the reservoir for the month (needed if IRESCO=1) (read in as  $m^3/s$  and converted to  $m^3/day$ ) ( $m^3/day$ )*
- integer, dimension(:), allocatable [parm::res\\_sub](#)

- number of subbasin reservoir is in (weather for the subbasin is used for the reservoir) (none)*
- integer, dimension(:,:), allocatable **parm::ires**
  - ires(1,:) beginning of mid-year nutrient settling "season" (none)*
  - ires(2,:) end of mid-year nutrient settling "season" (none)*
- integer, dimension(:), allocatable **parm::iresco**
  - outflow simulation code (none):*
  - 0 compute outflow for uncontrolled reservoir with average annual release rate*
  - 1 measured monthly outflow*
  - 2 simulated controlled outflow-target release*
  - 3 measured daily outflow*
  - 4 stage/volume/outflow relationship*
- integer, dimension(:), allocatable **parm::iyres**
  - year of the simulation that the reservoir becomes operational (none)*
- integer, dimension(:), allocatable **parm::mores**
  - month the reservoir becomes operational (none)*
- integer, dimension(:,:), allocatable **parm::iflodr**
  - iflodr(1,:) beginning month of non-flood season (needed if IRESCO=2) (none)*
  - iflodr(2,:) ending month of non-flood season (needed if IRESCO=2) (none)*
- integer, dimension(:), allocatable **parm::ndtargr**
  - number of days to reach target storage from current reservoir storage (needed if IRESCO=2) (days)*
- real \*8, dimension(:), allocatable **parm::ap\_ef**
  - application efficiency (0-1) (none)*
- real \*8, dimension(:), allocatable **parm::decay\_f**
  - exponential of the rate constant for degradation of the pesticide on foliage (none)*
- real \*8, dimension(:), allocatable **parm::skoc**
  - soil adsorption coefficient normalized for soil organic carbon content ((mg/kg)/(mg/L))*
- real \*8, dimension(:), allocatable **parm::decay\_s**
  - exponential of the rate constant for degradation of the pesticide in soil (none)*
- real \*8, dimension(:), allocatable **parm::pst\_wof**
  - fraction of pesticide on foliage which is washed-off by a rainfall event (none)*
- real \*8, dimension(:), allocatable **parm::pst\_wsol**
  - solubility of chemical in water (mg/L (ppm))*
- real \*8, dimension(:), allocatable **parm::irramt**
  - depth of irrigation water applied to HRU (mm H2O)*
- real \*8, dimension(:), allocatable **parm::phusw**
- integer, dimension(:), allocatable **parm::pstflg**
  - flag for types of pesticide used in watershed. Array location is pesticide ID number*
  - 0: pesticide not used*
  - 1: pesticide used*
- integer, dimension(:), allocatable **parm::nope**
  - sequence number of pesticide in NPNO(:) (none)*
- integer, dimension(:), allocatable **parm::nop**
- integer, dimension(:), allocatable **parm::yr\_skip**
- integer, dimension(:), allocatable **parm::icrmx**
- integer, dimension(:), allocatable **parm::nopmx**
- integer, dimension(:,:), allocatable **parm::mgtop**
- integer, dimension(:,:), allocatable **parm::idop**
- integer, dimension(:,:), allocatable **parm::mgt1iop**
- integer, dimension(:,:), allocatable **parm::mgt2iop**
- integer, dimension(:,:), allocatable **parm::mgt3iop**
- integer, dimension(:,:), allocatable **parm::mgt10iop**
- real \*8, dimension(:,:), allocatable **parm::mgt4op**
- real \*8, dimension(:,:), allocatable **parm::mgt5op**

- real \*8, dimension(:,:), allocatable **parm::mgt6op**
- real \*8, dimension(:,:), allocatable **parm::mgt7op**
- real \*8, dimension(:,:), allocatable **parm::mgt8op**
- real \*8, dimension(:,:), allocatable **parm::mgt9op**
- real \*8, dimension(:,:), allocatable **parm::phu\_op**
- real \*8, dimension(:), allocatable **parm::cnyld**  
*fraction of nitrogen in yield (kg N/kg yield)*
- real \*8, dimension(:), allocatable **parm::rsdco\_pl**  
*plant residue decomposition coefficient. The fraction of residue which will decompose in a day assuming optimal moisture, temperature, C:N ratio, and C:P ratio (none)*
- real \*8, dimension(:,:), allocatable **parm::wac2**  
*wac2(1,:) 1st shape parameter for radiation use efficiency equation (none)*  
*wac2(2,:) 2nd shape parameter for radiation use efficiency equation (none)*
- real \*8, dimension(:), allocatable **parm::alai\_min**  
*minimum LAI during winter dormant period ( $m^2/m^2$ )*
- real \*8, dimension(:,:), allocatable **parm::leaf**  
*leaf(1,:) 1st shape parameter for leaf area development equation (none)*  
*leaf(2,:) 2nd shape parameter for leaf area development equation (none)*
- real \*8, dimension(:), allocatable **parm::wsyf**  
*Value of harvest index between 0 and HVSTI which represents the lowest value expected due to water stress ((kg/ha)/(kg/ha))*
- real \*8, dimension(:), allocatable **parm::bio\_e**  
*biomass-energy ratio. The potential (unstressed) growth rate per unit of intercepted photosynthetically active radiation ((kg/ha)/(MJ/m\*\*2))*
- real \*8, dimension(:), allocatable **parm::hvsti**  
*harvest index: crop yield/aboveground biomass ((kg/ha)/(kg/ha))*
- real \*8, dimension(:), allocatable **parm::t\_base**  
*minimum temperature for plant growth (deg C)*
- real \*8, dimension(:), allocatable **parm::t\_opt**  
*optimal temperature for plant growth (deg C)*
- real \*8, dimension(:), allocatable **parm::chtmx**  
*maximum canopy height (m)*
- real \*8, dimension(:), allocatable **parm::cvm**  
*natural log of USLE\_C (the minimum value of the USLE C factor for the land cover) (none)*
- real \*8, dimension(:), allocatable **parm::gsi**  
*maximum stomatal conductance (m/s)*
- real \*8, dimension(:), allocatable **parm::vpd2**  
*rate of decline in stomatal conductance per unit increase in vapor pressure deficit ((m/s)\*(1/kPa))*
- real \*8, dimension(:), allocatable **parm::wavp**  
*rate of decline in radiation use efficiency as a function of vapor pressure deficit (none)*
- real \*8, dimension(:), allocatable **parm::bio\_leaf**  
*fraction of leaf/needle biomass that drops during dormancy (for trees only) (none)*
- real \*8, dimension(:), allocatable **parm::blai**  
*maximum (potential) leaf area index (none)*
- real \*8, dimension(:), allocatable **parm::cpyld**  
*fraction of phosphorus in yield (kg P/kg yield)*
- real \*8, dimension(:), allocatable **parm::dlai**  
*fraction of growing season when leaf area declines (none)*
- real \*8, dimension(:), allocatable **parm::rdmx**  
*maximum root depth of plant (m)*
- real \*8, dimension(:,:), allocatable **parm::bio\_n**

- bio\_n(1,:) 1st shape parameter for plant N uptake equation (none)*
- bio\_n(2,:) 2nd shape parameter for plant N uptake equation (none)*
- real \*8, dimension(:,:), allocatable [parm::bio\\_p](#)
  - bio\_p(1,:) 1st shape parameter for plant P uptake equation (none)*
  - bio\_p(2,:) 2nd shape parameter for plant P uptake equation (none)*
- real \*8, dimension(:), allocatable [parm::bm\\_dieoff](#)
  - fraction above ground biomass that dies off at dormancy (fraction)*
- real \*8, dimension(:), allocatable [parm::bmx\\_trees](#)
- real \*8, dimension(:), allocatable [parm::ext\\_coef](#)
- real \*8, dimension(:,:), allocatable [parm::rsr](#)
  - rsr(1,:) initial root to shoot ratio at the beg of growing season*
  - rsr(2,:) root to shoot ratio at the end of the growing season*
- real \*8, dimension(:), allocatable [parm::pltnfr1](#)
  - nitrogen uptake parameter #1: normal fraction of N in crop biomass at emergence (kg N/kg biomass)*
- real \*8, dimension(:), allocatable [parm::pltnfr3](#)
  - nitrogen uptake parameter #3: normal fraction of N in crop biomass at maturity (kg N/kg biomass)*
- real \*8, dimension(:), allocatable [parm::pltpfr1](#)
  - phosphorus uptake parameter #1: normal fraction of P in crop biomass at emergence (kg P/kg biomass)*
- real \*8, dimension(:), allocatable [parm::pltpfr3](#)
  - phosphorus uptake parameter #3: normal fraction of P in crop biomass at maturity (kg P/kg biomass)*
- integer, dimension(:), allocatable [parm::idc](#)
  - crop/landcover category (none):*
  - 1 warm season annual legume*
  - 2 cold season annual legume*
  - 3 perennial legume*
  - 4 warm season annual*
  - 5 cold season annual*
  - 6 perennial*
  - 7 trees*
- integer, dimension(:), allocatable [parm::mat\\_yrs](#)
- real \*8, dimension(:), allocatable [parm::bactpdb](#)
  - concentration of persistent bacteria in manure (fertilizer) (cfu/g manure)*
- real \*8, dimension(:), allocatable [parm::fminn](#)
  - fraction of fertilize/manure that is mineral nitrogen (NO3 + NH3) (kg minN/kg fert)*
- real \*8, dimension(:), allocatable [parm::forgn](#)
  - fraction of organic nitrogen in fertilizer/manure (kg orgN/kg fert)*
- real \*8, dimension(:), allocatable [parm::forgp](#)
  - fraction of fertilizer/manure that is organic phosphorus (kg orgP/kg fert)*
- real \*8, dimension(:), allocatable [parm::bactkddb](#)
  - fraction of bacteria in solution (the remaining fraction is sorbed to soil particles) (none):*
  - 1: all bacteria in solution*
  - 0: all bacteria sorbed to soil particles*
- real \*8, dimension(:), allocatable [parm::bactlpdb](#)
  - concentration of less persistent bacteria in manure (fertilizer) (cfu/g manure)*
- real \*8, dimension(:), allocatable [parm::fminp](#)
  - fraction of fertilizer that is mineral phosphorus in fertilizer/manure (kg minP/kg fert)*
- real \*8, dimension(:), allocatable [parm::fnh3n](#)
  - fraction of mineral N content that is NH3-N in fertilizer/manure (kg NH3-N/kg minN)*
- character(len=8), dimension(200) [parm::fertnm](#)
  - name of fertilizer*
- real \*8, dimension(:), allocatable [parm::curbden](#)
  - curb length density in HRU (km/ha)*
- real \*8, dimension(:), allocatable [parm::dirtmx](#)

- maximum amount of solids allowed to build up on impervious surfaces (kg/curb km)*
- real \*8, dimension(:), allocatable [parm::fimp](#)  
*fraction of HRU area that is impervious (both directly and indirectly connected) (fraction)*
- real \*8, dimension(:), allocatable [parm::urbcoef](#)  
*wash-off coefficient for removal of constituents from an impervious surface (1/mm)*
- real \*8, dimension(:), allocatable [parm::thalf](#)  
*time for the amount of solids on impervious areas to build up to 1/2 the maximum level (days)*
- real \*8, dimension(:), allocatable [parm::tnconc](#)  
*concentration of total nitrogen in suspended solid load from impervious areas (mg N/kg sed)*
- real \*8, dimension(:), allocatable [parm::tno3conc](#)  
*concentration of NO3-N in suspended solid load from impervious areas (mg NO3-N/kg sed)*
- real \*8, dimension(:), allocatable [parm::tpconc](#)  
*concentration of total phosphorus in suspended solid load from impervious areas (mg P/kg sed)*
- real \*8, dimension(:), allocatable [parm::fcimp](#)  
*fraction of HRU area that is classified as directly connected impervious (fraction)*
- real \*8, dimension(:), allocatable [parm::urbcn2](#)  
*SCS curve number for moisture condition II in impervious areas (none)*
- real \*8 [parm::fr\\_curb](#)  
*availability factor, the fraction of the curb length that is sweepable (none)*
- real \*8 [parm::frt\\_kg](#)  
*amount of fertilizer applied to HRU (kg/ha)*
- real \*8 [parm::pst\\_dep](#)  
*depth of pesticide in the soil (mm)*
- real \*8 [parm::sweepeff](#)  
*removal efficiency of sweeping operation (none)*
- real \*8, dimension(:), allocatable [parm::ranrns\\_hru](#)  
*random roughness for a given HRU (mm)*
- integer, dimension(:), allocatable [parm::itill](#)
- real \*8, dimension(:), allocatable [parm::deptil](#)  
*depth of mixing caused by tillage operation (mm)*
- real \*8, dimension(:), allocatable [parm::effmix](#)  
*mixing efficiency of tillage operation (none)*
- real \*8, dimension(:), allocatable [parm::ranrns](#)  
*random roughness of a given tillage operation (mm)*
- character(len=8), dimension(550) [parm::tillnm](#)  
*8-character name for the tillage operation*
- real \*8, dimension(:), allocatable [parm::rnum1s](#)  
*For ICODES equal to (none)*  
*0,1,3,5,9: not used*  
*2: fraction of overland flow in channel*  
*4: amount of water transferred (as defined by INUM4S)*  
*7,8,10,11: drainage area in square kilometers associated with the record file*  
*12: reparation coefficient.*
- real \*8, dimension(:), allocatable [parm::hyd\\_dakm](#)  
*total drainage area of hydrograph in square kilometers (km<sup>2</sup>)*
- real \*8, dimension(:,:), allocatable [parm::shyd](#)  
*shyd(1,:) water (m<sup>3</sup> H2O)*  
*shyd(2,:) sediment or suspended solid load (metric tons)*  
*shyd(3,:) organic nitrogen (kg N)*  
*shyd(4,:) organic phosphorus (kg P)*  
*shyd(5,:) nitrate (kg N)*  
*shyd(6,:) soluble phosphorus (kg P)*  
*shyd(7,:) soluble pesticides (kg P)*  
*shyd(8,:) sorbed pesticides (kg P)*



- real \*8, dimension(:,:), allocatable [parm::varoute](#)  
*varoute(:, :) daily routing storage array (varies):*  
*varoute(1,:) temperature (deg C)*  
*varoute(2,:) water ( $m^3 H_2O$ )*  
*varoute(3,:) sediment or suspended solid load (metric tons)*  
*varoute(4,:) organic nitrogen (kg N)*  
*varoute(5,:) organic phosphorus (kg P)*  
*varoute(6,:) nitrate (kg N)*  
*varoute(7,:) soluble mineral phosphorus (kg P)*  
*varoute(11,:) pesticide in solution (mg pst)*  
*varoute(12,:) pesticide sorbed to sediment (mg pst)*  
*varoute(13,:) chlorophyll-a (kg)*  
*varoute(14,:) ammonium (kg N)*  
*varoute(15,:) nitrite (kg N)*  
*varoute(16,:) carbonaceous biological oxygen demand (kg)*  
*varoute(17,:) dissolved oxygen (kg)*  
*varoute(18,:) persistent bacteria (# cfu/100ml)*  
*varoute(19,:) less persistent bacteria (# cfu/100ml)* *varoute(20,:) conservative metal #1 (kg)* *varoute(21,:) conservative metal #2 (kg)* *varoute(22,:) conservative metal #3 (kg)*
- real \*8, dimension(:,:), allocatable [parm::vartran](#)
- real \*8, dimension(:,:), allocatable [parm::hhvaroute](#)  
*routing storage array for hourly time step (varies)*  
*hhvaroute(1,:,:) temperature (deg C)*  
*hhvaroute(2,:,:) water ( $m^3 H_2O$ )*  
*hhvaroute(3,:,:) sediment or suspended solid load (metric tons)*  
*hhvaroute(4,:,:) organic nitrogen (kg N)*  
*hhvaroute(5,:,:) organic phosphorus (kg P)*  
*hhvaroute(6,:,:) nitrate (kg N)*  
*hhvaroute(7,:,:) soluble mineral phosphorus (kg P)*  
*hhvaroute(11,:,:) pesticide in solution (mg pst)*  
*hhvaroute(12,:,:) pesticide sorbed to sediment (mg pst)*  
*hhvaroute(13,:,:) chlorophyll-a (kg)*  
*hhvaroute(14,:,:) ammonium (kg N)*  
*hhvaroute(15,:,:) nitrite (kg N)*  
*hhvaroute(16,:,:) carbonaceous biological oxygen demand (kg)*  
*hhvaroute(17,:,:) dissolved oxygen (kg O2)*  
*hhvaroute(18,:,:) persistent bacteria (# cfu/100ml)*  
*hhvaroute(19,:,:) less persistent bacteria (# cfu/100ml)*  
*hhvaroute(20,:,:) conservative metal #1 (kg)*  
*hhvaroute(21,:,:) conservative metal #2 (kg)*  
*hhvaroute(22,:,:) conservative metal #3 (kg)*
- integer, dimension(:), allocatable [parm::icodes](#)  
*routing command code (none):*  
*0 = finish*  
*1 = subbasin*  
*2 = route*  
*3 = routres*  
*4 = transfer*  
*5 = add*  
*6 = rechour*  
*7 = recmon*  
*8 = recyear*  
*9 = save*  
*10 = recday*  
*11 = recnst*  
*12 = structure*  
*13 = apex*  
*14 = saveconc*  
*15 =*  
*16 = autocal*  
*17 = routing unit*
- integer, dimension(:), allocatable [parm::ihouts](#)



*For ICODES equal to (none)*

*0: not used*

*1,2,3,5,6,7,8,10,11: hydrograph storage location number*

*4: departure type (1=reach, 2=reservoir)*

*9: hydrograph storage location of data to be printed to event file*

*14: hydrograph storage location of data to be printed to saveconc file.*

- integer, dimension(:), allocatable **parm::inum1s**

*For ICODES equal to (none)*

*0: not used*

*1: subbasin number*

*2: reach number*

*3: reservoir number*

*4: reach or res # flow is diverted from*

*5: hydrograph storage location of 1st dataset to be added*

*6,7,8,9,10,11,14: file number.*

- integer, dimension(:), allocatable **parm::inum2s**

*For ICODES equal to (none)*

*0,1,7,8,10,11: not used*

*2,3: inflow hydrograph storage location*

*4: destination type (1=reach, 2=reservoir)*

*5: hydrograph storage location of 2nd dataset to be added*

*9,14: print frequency (0=daily, 1=hourly)*

- integer, dimension(:), allocatable **parm::inum3s**

*For ICODES equal to (none)*

*0,1,5,7,8,10,11: not used*

*2,3: subbasin number 4: destination number. Reach or reservoir receiving water*

*9: print format (0=normal, fixed format; 1=txt format for AV interface, recday)*

- integer, dimension(:), allocatable **parm::inum4s**

*For ICODES equal to (none)*

*0,2,3,5,7,8,9,10,11: not used*

*1: GIS code printed to output file (optional)*

*4: rule code governing transfer of water (1=fraction transferred out, 2=min volume or flow left, 3=exact amount transferred)*

- integer, dimension(:), allocatable **parm::inum5s**
- integer, dimension(:), allocatable **parm::inum6s**
- integer, dimension(:), allocatable **parm::inum7s**
- integer, dimension(:), allocatable **parm::inum8s**
- integer, dimension(:), allocatable **parm::subed**
- character(len=10), dimension(:), allocatable **parm::recmonps**
- character(len=10), dimension(:), allocatable **parm::reccnstps**
- character(len=5), dimension(:), allocatable **parm::subnum**
- character(len=4), dimension(:), allocatable **parm::hruno**
- real \*8, dimension(:), allocatable **parm::grwat\_n**  
*Mannings's n for grassed waterway (none)*
- integer, dimension(:), allocatable **parm::grwat\_i**  
*flag for the simulation of grass waterways (none)*  
*= 0 inactive*  
*= 1 active*
- real \*8, dimension(:), allocatable **parm::grwat\_l**  
*length of grass waterway (km)*
- real \*8, dimension(:), allocatable **parm::grwat\_w**  
*average width of grassed waterway (m)*
- real \*8, dimension(:), allocatable **parm::grwat\_d**  
*depth of grassed waterway from top of bank to bottom (m)*
- real \*8, dimension(:), allocatable **parm::grwat\_s**  
*average slope of grassed waterway channel (m)*
- real \*8, dimension(:), allocatable **parm::grwat\_spcon**

- linear parameter defined by user for calculating sediment transport in grassed waterways (none)*
- real \*8, dimension(:), allocatable **parm::tc\_gwat**  
*time of concentration for grassed waterway and its drainage area (none)*
- real \*8, dimension(:), allocatable **parm::pot\_tilemm**
- real \*8, dimension(:), allocatable **parm::pot\_volxmm**
- real \*8, dimension(:), allocatable **parm::pot\_fr**  
*fraction of HRU area that drains into pothole ( $\text{km}^2/\text{km}^2$ )*
- real \*8, dimension(:), allocatable **parm::pot\_vol**  
*initial or current volume of water stored in the depression/impounded area (read in as mm and converted to  $\text{m}^3$ ) (needed only if current HRU is IPOT) (mm or  $\text{m}^3 \text{H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable **parm::potsa**  
*surface area of impounded water body (ha)*
- real \*8, dimension(:), allocatable **parm::wfsH**  
*wetting front matric potential (average capillary suction at wetting front) (mm)*
- real \*8, dimension(:), allocatable **parm::potflwi**  
*water entering pothole on day ( $\text{m}^3 \text{H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable **parm::potsedi**  
*sediment entering pothole on day (metric tons)*
- real \*8, dimension(:), allocatable **parm::newrti**  
*infiltration rate for last time step from the previous day (mm/hr)*
- real \*8, dimension(:), allocatable **parm::fsred**  
*reduction in bacteria loading from filter strip (none)*
- real \*8, dimension(:), allocatable **parm::pot\_no3**  
*amount of nitrate in pothole water body (kg N)*
- real \*8, dimension(:), allocatable **parm::pot\_sed**  
*amount of sediment in pothole water body (metric tons)*
- real \*8, dimension(:), allocatable **parm::dis\_stream**  
*average distance to stream (m)*
- real \*8, dimension(:), allocatable **parm::sed\_con**
- real \*8, dimension(:), allocatable **parm::orgn\_con**
- real \*8, dimension(:), allocatable **parm::orgp\_con**
- real \*8, dimension(:), allocatable **parm::pot\_k**  
*hydraulic conductivity of soil surface of pothole defaults to conductivity of upper soil (0.↔ 01-10.) layer*
- real \*8, dimension(:), allocatable **parm::soln\_con**
- real \*8, dimension(:), allocatable **parm::solp\_con**
- real \*8, dimension(:), allocatable **parm::n\_reduc**  
*nitrogen uptake reduction factor (not currently used; defaulted 300.)*
- real \*8, dimension(:), allocatable **parm::n\_in**  
*power function exponent for calculating nitrate concentration in subsurface drains (1.0 - 3.0) (dimensionless)*
- real \*8, dimension(:), allocatable **parm::n\_inco**  
*coefficient for power function for calculating nitrate concentration in subsurface drains (0.5 - 4.0) (dimensionless)*
- integer, dimension(:), allocatable **parm::ioper**
- real \*8, dimension(:), allocatable **parm::usle\_ls**  
*USLE equation length slope (LS) factor (none)*
- real \*8, dimension(:), allocatable **parm::filterw**  
*filter strip width for bacteria transport (m)*
- real \*8, dimension(:), allocatable **parm::phuacc**  
*fraction of plant heat units accumulated (none)*
- real \*8, dimension(:), allocatable **parm::sumix**  
*sum of all tillage mixing efficiencies for HRU operation (none)*
- real \*8, dimension(:), allocatable **parm::epco**

- plant water uptake compensation factor (0-1) (none)*

  - real \*8, dimension(:), allocatable [parm::esco](#)
- soil evaporation compensation factor (0-1) (none)*

  - real \*8, dimension(:), allocatable [parm::hru\\_slp](#)
- average slope steepness in HRU (m/m)*

  - real \*8, dimension(:), allocatable [parm::slsubbsn](#)
- average slope length for subbasin (m)*

  - real \*8, dimension(:), allocatable [parm::erorgn](#)
- organic N enrichment ratio, if left blank the model will calculate for every event (none)*

  - real \*8, dimension(:), allocatable [parm::erorgp](#)
- organic P enrichment ratio, if left blank the model will calculate for every event (none)*

  - real \*8, dimension(:), allocatable [parm::biomix](#)
- biological mixing efficiency. Mixing of soil due to activity of earthworms and other soil biota. Mixing is performed at the end of every calendar year (none)*

  - real \*8, dimension(:), allocatable [parm::pnd\\_seci](#)
- secchi-disk depth of pond (m)*

  - real \*8, dimension(:), allocatable [parm::canmx](#)
- maximum canopy storage (mm H2O)*

  - real \*8, dimension(:), allocatable [parm::divmax](#)
- maximum daily irrigation diversion from the reach (when IRRSC=1 or IRR=3): when value is positive the units are mm H2O; when the value is negative, the units are (10<sup>4</sup> m<sup>3</sup> H2O) (mm H2O or 10<sup>4</sup> m<sup>3</sup> H2O)*

  - real \*8, dimension(:), allocatable [parm::flowmin](#)
- minimum instream flow for irrigation diversions when IRRSC=1, irrigation water will be diverted only when streamflow is at or above FLOWMIN (m<sup>3</sup>/s)*

  - real \*8, dimension(:), allocatable [parm::usle\\_p](#)
- USLE equation support practice (P) factor (none)*

  - real \*8, dimension(:), allocatable [parm::lat\\_sed](#)
- sediment concentration in lateral flow (g/L)*

  - real \*8, dimension(:), allocatable [parm::rch\\_dakm](#)
- total drainage area contributing to flow at the outlet (pour point) of the reach in square kilometers (km<sup>2</sup>)*

  - real \*8, dimension(:), allocatable [parm::cn](#)
- cn(1,:) SCS runoff curve number for moisture condition I (none)*  
*cn(2,:) SCS runoff curve number for moisture condition II (none)*  
*cn(3,:) SCS runoff curve number for moisture condition III (none)*

  - real \*8, dimension(:), allocatable [parm::pnd\\_no3s](#)
- amount of nitrate originating from lateral flow in pond at end of day or at beginning of day(kg N)*

  - real \*8, dimension(:), allocatable [parm::lat\\_ttime](#)
- lateral flow travel time or exponential of the lateral flow travel time (days or none)*

  - real \*8, dimension(:), allocatable [parm::flowfr](#)
- fraction of available flow in reach that is allowed to be applied to the HRU (none)*

  - real \*8, dimension(:), allocatable [parm::sol\\_zmx](#)
- maximum rooting depth (mm)*

  - real \*8, dimension(:), allocatable [parm::tile\\_ttime](#)
- exponential of the tile flow travel time (none)*

  - real \*8, dimension(:), allocatable [parm::slsoil](#)
- slope length for lateral subsurface flow (m)*

  - real \*8, dimension(:), allocatable [parm::gwminp](#)
- soluble P concentration in groundwater loading to reach (mg P/L)*

  - real \*8, dimension(:), allocatable [parm::sol\\_cov](#)
- amount of residue on soil surface (kg/ha)*

  - real \*8, dimension(:), allocatable [parm::sed\\_stl](#)
- fraction of sediment remaining suspended in impoundment after settling for one day (kg/kg)*

- real \*8, dimension(:), allocatable [parm::ov\\_n](#)  
*Manning's "n" value for overland flow (none)*
- real \*8, dimension(:), allocatable [parm::pnd\\_no3](#)  
*amount of nitrate originating from surface runoff in pond at end of day or at beginning of day (kg N)*
- real \*8, dimension(:), allocatable [parm::pnd\\_solp](#)  
*amount of soluble P originating from surface runoff in pond at end of day or at beginning of day (kg P)*
- real \*8, dimension(:), allocatable [parm::yldanu](#)  
*annual yield (dry weight) in the HRU (metric tons/ha)*
- real \*8, dimension(:), allocatable [parm::pnd\\_orgn](#)  
*amount of organic N originating from surface runoff in pond at end of day or at beginning of day (kg N)*
- real \*8, dimension(:), allocatable [parm::pnd\\_orgp](#)  
*amount of organic P originating from surface runoff in pond at end of day or at beginning of day (kg P)*
- real \*8, dimension(:), allocatable [parm::twlpnd](#)  
*water lost through seepage from ponds on day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::twlwet](#)  
*water lost through seepage from wetlands on day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::hru\\_fr](#)  
*fraction of subbasin area contained in HRU ( $\text{km}^2/\text{km}^2$ )*
- real \*8, dimension(:), allocatable [parm::sol\\_sumul](#)  
*amount of water held in soil profile at saturation (mm H2O)*
- real \*8, dimension(:), allocatable [parm::pnd\\_chla](#)  
*amount of chlorophyll-a in pond at end of day (kg chl\_a)*
- real \*8, dimension(:), allocatable [parm::hru\\_km](#)  
*area of HRU in square kilometers ( $\text{km}^2$ )*
- real \*8, dimension(:), allocatable [parm::bio\\_ms](#)  
*land cover/crop biomass (dry weight) (kg/ha)*
- real \*8, dimension(:), allocatable [parm::sol\\_alb](#)  
*albedo when soil is moist (none)*
- real \*8, dimension(:), allocatable [parm::strsw](#)  
*fraction of potential plant growth achieved on the day where the reduction is caused by water stress (none)*
- real \*8, dimension(:), allocatable [parm::pnd\\_fr](#)  
*fraction of HRU/subbasin area that drains into ponds (none)*
- real \*8, dimension(:), allocatable [parm::pnd\\_k](#)  
*hydraulic conductivity through bottom of ponds (mm/hr)*
- real \*8, dimension(:), allocatable [parm::pnd\\_psa](#)  
*surface area of ponds when filled to principal spillway (ha)*
- real \*8, dimension(:), allocatable [parm::pnd\\_pvol](#)  
*runoff volume of water from catchment area needed to fill the ponds to the principal spillway (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [parm::pnd\\_esa](#)  
*surface area of ponds when filled to emergency spillway (ha)*
- real \*8, dimension(:), allocatable [parm::pnd\\_evol](#)  
*runoff volume of water from catchment area needed to fill the ponds to the emergency spillway (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [parm::pnd\\_vol](#)  
*volume of water in ponds (UNIT CHANGE!) ( $10^4 \text{ m}^3 \text{ H}_2\text{O}$  or  $\text{m}^3 \text{ H}_2\text{O}$ )*
- real \*8, dimension(:), allocatable [parm::yldaa](#)  
*average annual yield (dry weight) in the HRU (metric tons)*
- real \*8, dimension(:), allocatable [parm::pnd\\_nsed](#)  
*normal sediment concentration in pond water (UNIT CHANGE!) (mg/kg or kg/kg)*
- real \*8, dimension(:), allocatable [parm::pnd\\_sed](#)

- sediment concentration in pond water (UNIT CHANGE!) (mg/kg or kg/kg)*
- real \*8, dimension(:), allocatable [parm::dep\\_imp](#)  
*depth to impervious layer (mm)*
- real \*8, dimension(:), allocatable [parm::strsa](#)
- real \*8, dimension(:), allocatable [parm::evpnd](#)
- real \*8, dimension(:), allocatable [parm::evwet](#)
- real \*8, dimension(:), allocatable [parm::wet\\_fr](#)  
*fraction of HRU/subbasin area that drains into wetlands (none)*
- real \*8, dimension(:), allocatable [parm::wet\\_k](#)  
*hydraulic conductivity of bottom of wetlands (mm/hr)*
- real \*8, dimension(:), allocatable [parm::wet\\_nsa](#)  
*surface area of wetlands in subbasin at normal water level (ha)*
- real \*8, dimension(:), allocatable [parm::wet\\_nvol](#)  
*runoff volume of water from catchment area needed to fill wetlands to normal water level (UNIT CHANGE!) ( $10^4$   $m^3$  H<sub>2</sub>O or  $m^3$  H<sub>2</sub>O)*
- integer, dimension(:), allocatable [parm::iwetgw](#)
- integer, dimension(:), allocatable [parm::iwetile](#)
- real \*8, dimension(:), allocatable [parm::wet\\_mxsa](#)  
*surface area of wetlands at maximum water level (ha)*
- real \*8, dimension(:), allocatable [parm::wet\\_mxvol](#)  
*runoff volume of water from catchment area needed to fill wetlands to maximum water level (UNIT CHANGE!) ( $10^4$   $m^3$  H<sub>2</sub>O or  $m^3$  H<sub>2</sub>O)*
- real \*8, dimension(:), allocatable [parm::wet\\_vol](#)  
*volume of water in wetlands (UNIT CHANGE!) ( $10^4$   $m^3$  H<sub>2</sub>O or  $m^3$  H<sub>2</sub>O)*
- real \*8, dimension(:), allocatable [parm::wet\\_nsed](#)  
*normal sediment concentration in wetland water (UNIT CHANGE!) (mg/kg or kg/kg)*
- real \*8, dimension(:), allocatable [parm::wet\\_sed](#)  
*sediment concentration in wetland water (UNIT CHANGE!) (mg/L or kg/L)*
- real \*8, dimension(:,:), allocatable [parm::bp](#)  
*bp(1,:) 1st shape parameter for the pond surface area equation (none)*  
*bp(2,:) 2nd shape parameter for the pond surface area equation (none)*
- real \*8, dimension(:), allocatable [parm::sci](#)  
*retention coefficient for CN method based on plant ET (none)*
- real \*8, dimension(:), allocatable [parm::smx](#)  
*retention coefficient for CN method based on soil moisture (none)*
- real \*8, dimension(:,:), allocatable [parm::bw](#)  
*bw(1,:) 1st shape parameter for the wetland surface area equation (none)*  
*bw(2,:) 2nd shape parameter for the wetland surface area equation (none)*
- real \*8, dimension(:), allocatable [parm::bactpq](#)  
*persistent bacteria in soil solution (# cfu/ $m^2$ )*
- real \*8, dimension(:), allocatable [parm::cnday](#)  
*curve number for current day, HRU and at current soil moisture (none)*
- real \*8, dimension(:), allocatable [parm::bactlp\\_plt](#)  
*less persistent bacteria on foliage (# cfu/ $m^2$ )*
- real \*8, dimension(:), allocatable [parm::bactp\\_plt](#)  
*persistent bacteria on foliage (# cfu/ $m^2$ )*
- real \*8, dimension(:), allocatable [parm::auto\\_eff](#)  
*fertilizer application efficiency calculated as the amount of N applied divided by the amount of N removed at harvest (none)*
- real \*8, dimension(:), allocatable [parm::secciw](#)  
*water clarity coefficient for wetland (none)*
- real \*8, dimension(:), allocatable [parm::sol\\_sw](#)

- amount of water stored in soil profile at end of any given day (mm H2O)*

  - real \*8, dimension(:), allocatable [parm::bactlpq](#)  
*less persistent bacteria in soil solution (# cfu/m<sup>2</sup>)*
  - real \*8, dimension(:), allocatable [parm::chlaw](#)  
*chlorophyll-a production coefficient for wetland (none)*
  - real \*8, dimension(:), allocatable [parm::tmpav](#)  
*average air temperature on current day in HRU (deg C)*
  - real \*8, dimension(:), allocatable [parm::bactlps](#)  
*less persistent bacteria attached to soil particles (# cfu/m<sup>2</sup>)*
  - real \*8, dimension(:), allocatable [parm::bactps](#)  
*persistent bacteria attached to soil particles (# cfu/m<sup>2</sup>)*
  - real \*8, dimension(:), allocatable [parm::sno\\_hru](#)  
*amount of water stored as snow in HRU on current day (mm H2O)*
  - real \*8, dimension(:), allocatable [parm::wet\\_orgn](#)  
*amount of organic N originating from surface runoff in wetland at end of day (kg N)*
  - real \*8, dimension(:), allocatable [parm::hru\\_ra](#)  
*solar radiation for the day in HRU (MJ/m<sup>2</sup>)*
  - real \*8, dimension(:), allocatable [parm::subp](#)  
*precipitation for the day in HRU (mm H2O)*
  - real \*8, dimension(:), allocatable [parm::rsdin](#)  
*initial residue cover (kg/ha)*
  - real \*8, dimension(:), allocatable [parm::tmn](#)  
*minimum air temperature on current day in HRU (deg C)*
  - real \*8, dimension(:), allocatable [parm::tmx](#)  
*maximum air temperature on current day in HRU (deg C)*
  - real \*8, dimension(:), allocatable [parm::tmp\\_hi](#)  
*last maximum temperature in HRU (deg C)*
  - real \*8, dimension(:), allocatable [parm::tmp\\_lo](#)  
*last minimum temperature in HRU (deg C)*
  - real \*8, dimension(:), allocatable [parm::usle\\_k](#)  
*USLE equation soil erodibility (K) factor (none)*
  - real \*8, dimension(:), allocatable [parm::tconc](#)  
*time of concentration for HRU (hour)*
  - real \*8, dimension(:), allocatable [parm::hru\\_rmx](#)  
*maximum possible solar radiation for the day in HRU (MJ/m<sup>2</sup>)*
  - real \*8, dimension(:), allocatable [parm::rwt](#)  
*fraction of total plant biomass that is in roots (none)*
  - real \*8, dimension(:), allocatable [parm::olai](#)
  - real \*8, dimension(:), allocatable [parm::usle\\_cfac](#)
  - real \*8, dimension(:), allocatable [parm::usle\\_eifac](#)
  - real \*8, dimension(:), allocatable [parm::sol\\_sumfc](#)  
*amount of water held in soil profile at field capacity (mm H2O)*
  - real \*8, dimension(:), allocatable [parm::t\\_ov](#)  
*time for flow from farthest point in subbasin to enter a channel (hour)*
  - real \*8, dimension(:), allocatable [parm::anano3](#)  
*total amount of NO3 applied during the year in auto-fertilization (kg N/ha)*
  - real \*8, dimension(:), allocatable [parm::aird](#)  
*amount of water applied to HRU on current day (mm H2O)*
  - real \*8, dimension(:), allocatable [parm::wet\\_orgp](#)  
*amount of organic P originating from surface runoff in wetland at end of day (kg P)*
  - real \*8, dimension(:), allocatable [parm::usle\\_mult](#)

- product of USLE  $K, P, LS, exp(rock)$  (none)*
- real \*8, dimension(:), allocatable [parm::rhd](#)  
*relative humidity for the day in HRU (none)*
- real \*8, dimension(:), allocatable [parm::u10](#)  
*wind speed (measured at 10 meters above surface) for the day in HRU (m/s)*
- real \*8, dimension(:), allocatable [parm::cht](#)  
*canopy height (m)*
- real \*8, dimension(:), allocatable [parm::aairr](#)  
*average annual amount of irrigation water applied to HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::lai\\_aamx](#)  
*maximum leaf area index for the entire period of simulation in the HRU (none)*
- real \*8, dimension(:), allocatable [parm::deepirr](#)  
*amount of water removed from deep aquifer for irrigation (mm H2O)*
- real \*8, dimension(:), allocatable [parm::shallirr](#)  
*amount of water removed from shallow aquifer for irrigation (mm H2O)*
- real \*8, dimension(:), allocatable [parm::wet\\_no3](#)  
*amount of nitrate originating from surface runoff in wetland at end of day (kg N)*
- real \*8, dimension(:), allocatable [parm::ovrlnd](#)  
*overland flow onto HRU from upstream routing unit (mm H2O)*
- real \*8, dimension(:), allocatable [parm::canstor](#)  
*amount of water held in canopy storage (mm H2O)*
- real \*8, dimension(:), allocatable [parm::irr\\_mx](#)  
*maximum irrigation amount per auto application (mm)*
- real \*8, dimension(:), allocatable [parm::auto\\_wstr](#)  
*water stress factor which triggers auto irrigation (none or mm)*
- integer, dimension(:), allocatable [parm::cfrt\\_id](#)  
*fertilizer/manure identification number from database (fert.dat) (none)*
- real \*8, dimension(:), allocatable [parm::cfrt\\_kg](#)  
*amount of fertilizer/manure applied to HRU on a given day ((kg/ha)/day)*
- integer, dimension(:), allocatable [parm::cpst\\_id](#)
- real \*8, dimension(:), allocatable [parm::cpst\\_kg](#)
- real \*8, dimension(:), allocatable [parm::irr\\_asq](#)  
*surface runoff ratio*
- real \*8, dimension(:), allocatable [parm::irr\\_eff](#)
- real \*8, dimension(:), allocatable [parm::irrsq](#)  
*surface runoff ratio (0-1) .1 is 10% surface runoff (frac)*
- real \*8, dimension(:), allocatable [parm::irrefm](#)
- real \*8, dimension(:), allocatable [parm::bio\\_eat](#)  
*dry weight of biomass removed by grazing daily ((kg/ha)/day)*
- real \*8, dimension(:), allocatable [parm::bio\\_trmp](#)  
*dry weight of biomass removed by trampling daily ((kg/ha)/day)*
- integer, dimension(:), allocatable [parm::ipst\\_freq](#)  
*number of days between applications (days)*
- integer, dimension(:), allocatable [parm::ifrt\\_freq](#)  
*number of days between applications in continuous fertilizer operation (days)*
- integer, dimension(:), allocatable [parm::irr\\_noa](#)
- integer, dimension(:), allocatable [parm::irr\\_sc](#)
- integer, dimension(:), allocatable [parm::irr\\_no](#)
- integer, dimension(:), allocatable [parm::imp\\_trig](#)  
*release/impound action code (none):*  
*0 begin impounding water*  
*1 release impounded water*



- integer, dimension(:), allocatable [parm::fert\\_days](#)  
*number of days continuous fertilization will be simulated (none)*
- integer, dimension(:), allocatable [parm::irr\\_sca](#)
- integer, dimension(:), allocatable [parm::idplt](#)  
*land cover/crop identification code for first crop grown in HRU (the only crop if there is no rotation) (from crop.dat) (none)*
- integer, dimension(:), allocatable [parm::wstrs\\_id](#)  
*water stress identifier (none):*  
*1 plant water demand*  
*2 soil water deficit*
- integer, dimension(:), allocatable [parm::pest\\_days](#)
- real \*8, dimension(:,:), allocatable [parm::bio\\_aahv](#)  
*harvested biomass of plant (kg/ha)*
- real \*8, dimension(:), allocatable [parm::wet\\_solp](#)  
*amount of soluble P originating from surface runoff in wetland at end of day (kg P)*
- real \*8, dimension(:), allocatable [parm::wet\\_chla](#)  
*amount of chlorophyll-a in wetland at end of day (kg chla)*
- real \*8, dimension(:), allocatable [parm::wet\\_no3s](#)  
*amount of nitrate originating from lateral flow in wetland at end of day (kg N)*
- real \*8, dimension(:), allocatable [parm::pstsol](#)  
*amount of soluble pesticide leached from bottom of soil profile on current day (kg pst/ha)*
- real \*8, dimension(:), allocatable [parm::pnd\\_no3g](#)  
*amount of nitrate originating from groundwater in pond at end of day or at beginning of day (kg N)*
- real \*8, dimension(:), allocatable [parm::wet\\_seci](#)  
*secchi-disk depth in wetland at end of day (m)*
- real \*8, dimension(:), allocatable [parm::delay](#)  
*groundwater delay: time required for water leaving the bottom of the root zone to reach the shallow aquifer (days)*
- real \*8, dimension(:), allocatable [parm::gwht](#)  
*groundwater height (m)*
- real \*8, dimension(:), allocatable [parm::gw\\_q](#)  
*groundwater contribution to streamflow from HRU on current day (mm H2O)*
- real \*8, dimension(:), allocatable [parm::pnd\\_solpg](#)  
*amount of soluble P originating from groundwater in pond at end of day or at beginning of day (kg P)*
- real \*8, dimension(:), allocatable [parm::alpha\\_bf](#)  
*alpha factor for groundwater recession curve (1/days)*
- real \*8, dimension(:), allocatable [parm::alpha\\_bfe](#)  
 $\exp(-\alpha_{bf})$  (none)
- real \*8, dimension(:), allocatable [parm::gw\\_spyld](#)  
*specific yield for shallow aquifer ( $m^3/m^3$ )*
- real \*8, dimension(:), allocatable [parm::alpha\\_bfe\\_d](#)  
 $\exp(-\alpha_{bf_d})$  (with  $\alpha_{bf_d}$  the alpha factor for groundwater recession curve of the deep aquifer (1/days)) (none)
- real \*8, dimension(:), allocatable [parm::gw\\_qdeep](#)  
*groundwater contribution to streamflow from deep aquifer from HRU on current day (mm H2O)*
- real \*8, dimension(:), allocatable [parm::gw\\_delaye](#)  
 $\exp(-1/\text{delay})$  where  $\text{delay}(:)$  is the groundwater delay (time required for water leaving the bottom of the root zone to reach the shallow aquifer; units-days) (none)
- real \*8, dimension(:), allocatable [parm::gw\\_revap](#)  
*revap coeff: this variable controls the amount of water moving from the shallow aquifer to the root zone as a result of soil moisture depletion (none)*
- real \*8, dimension(:), allocatable [parm::rchrg\\_dp](#)  
*recharge to deep aquifer: the fraction of root zone percolation that reaches the deep aquifer (none)*



- real \*8, dimension(:), allocatable [parm::anion\\_excl](#)  
*fraction of porosity from which anions are excluded*
- real \*8, dimension(:), allocatable [parm::revapmn](#)  
*threshold depth of water in shallow aquifer required to allow revap to occur (mm H2O)*
- real \*8, dimension(:), allocatable [parm::rchrg](#)  
*amount of water recharging both aquifers on current day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::bio\\_min](#)  
*minimum plant biomass for grazing (kg/ha)*
- real \*8, dimension(:), allocatable [parm::ffc](#)  
*initial HRU soil water content expressed as fraction of field capacity (none)*
- real \*8, dimension(:), allocatable [parm::surqsolp](#)  
*amount of soluble phosphorus in surface runoff in HRU for the day (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::deepst](#)  
*depth of water in deep aquifer (mm H2O)*
- real \*8, dimension(:), allocatable [parm::shallst](#)  
*depth of water in shallow aquifer in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::wet\\_solpg](#)  
*amount of soluble P originating from groundwater in wetland at end of day (kg P)*
- real \*8, dimension(:), allocatable [parm::cklsp](#)
- real \*8, dimension(:), allocatable [parm::rchrg\\_src](#)
- real \*8, dimension(:), allocatable [parm::trapeff](#)  
*filter strip trapping efficiency (used for everything but bacteria) (none)*
- real \*8, dimension(:), allocatable [parm::sol\\_avbd](#)  
*average bulk density for soil profile (Mg/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable [parm::wet\\_no3g](#)  
*amount of nitrate originating from groundwater in wetland at end of day (kg N)*
- real \*8, dimension(:), allocatable [parm::tdrain](#)  
*time to drain soil to field capacity yield used in autofertilization (hours)*
- real \*8, dimension(:), allocatable [parm::gwqmn](#)  
*threshold depth of water in shallow aquifer required before groundwater flow will occur (mm H2O)*
- real \*8, dimension(:), allocatable [parm::snotmp](#)  
*temperature of snow pack in HRU (deg C)*
- real \*8, dimension(:), allocatable [parm::pplnt](#)  
*plant uptake of phosphorus in HRU for the day (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::gdrain](#)  
*drain tile lag time: the amount of time between the transfer of water from the soil to the drain tile and the release of the water from the drain tile to the reach (hours)*
- real \*8, dimension(:), allocatable [parm::ddrain](#)  
*depth of drain tube from the soil surface (mm)*
- real \*8, dimension(:), allocatable [parm::sol\\_crk](#)  
*crack volume potential of soil (none)*
- real \*8, dimension(:), allocatable [parm::brt](#)  
*fraction of surface runoff within the subbasin which takes 1 day or less to reach the subbasin outlet (none)*
- real \*8, dimension(:), allocatable [parm::dayl](#)  
*length of the current day in HRU (hours)*
- real \*8, dimension(:), allocatable [parm::sstmaxd](#)  
*static maximum depression storage; read from .sdr (mm)*
- real \*8, dimension(:), allocatable [parm::re](#)  
*effective radius of drains (mm)*
- real \*8, dimension(:), allocatable [parm::sdrain](#)  
*distance between two drain tubes or tiles (mm)*

- real \*8, dimension(:), allocatable [parm::drain\\_co](#)  
*drainage coefficient (mm/day)*
- real \*8, dimension(:), allocatable [parm::latksatf](#)  
*multiplication factor to determine conk(j1,j) from sol\_k(j1,j) for HRU (none)*
- real \*8, dimension(:), allocatable [parm::pc](#)  
*pump capacity (default pump capacity = 1.042mm/hr or 25mm/day) (mm/hr)*
- real \*8, dimension(:), allocatable [parm::stmaxd](#)  
*maximum surface depressional storage for day in a given HRU (mm)*
- real \*8, dimension(:), allocatable [parm::rnd3](#)  
*random number between 0.0 and 1.0 (none)*
- real \*8, dimension(:), allocatable [parm::rnd2](#)  
*random number between 0.0 and 1.0 (none)*
- real \*8, dimension(:), allocatable [parm::twash](#)  
*time that solids have built-up on streets (days)*
- real \*8, dimension(:), allocatable [parm::doxq](#)  
*dissolved oxygen concentration in the surface runoff on current day in HRU (mg/L)*
- real \*8, dimension(:), allocatable [parm::sol\\_cnsw](#)  
*amount of water stored in soil profile used to calculate daily CN value (initial soil water content for day) (mm H2O)*
- real \*8, dimension(:), allocatable [parm::rnd8](#)  
*random number between 0.0 and 1.0 (none)*
- real \*8, dimension(:), allocatable [parm::rnd9](#)  
*random number between 0.0 and 1.0 (none)*
- real \*8, dimension(:), allocatable [parm::percn](#)  
*amount of nitrate percolating past bottom of soil profile during the day (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::sol\\_sumwp](#)
- real \*8, dimension(:), allocatable [parm::qdr](#)  
*total or net amount of water entering main channel for day from HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::cbodu](#)  
*amount of N applied in autofert operation in year (kg N/ha) (mg/L)*
- real \*8, dimension(:), allocatable [parm::chl\\_a](#)  
*chlorophyll-a concentration in water yield on current day in HRU (microgram/L)*
- real \*8, dimension(:), allocatable [parm::latq](#)  
*total amount of water in lateral flow in soil profile for the day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::nplnt](#)  
*plant uptake of nitrogen in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::latno3](#)  
*amount of nitrate transported with lateral flow in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::minpgw](#)  
*soluble P loading to reach in groundwater (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::no3gw](#)  
*nitrate loading to reach in groundwater (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::tileq](#)
- real \*8, dimension(:), allocatable [parm::tileno3](#)
- real \*8, dimension(:), allocatable [parm::sedorgn](#)  
*amount of organic nitrogen in surface runoff in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::sedminpa](#)  
*amount of active mineral phosphorus sorbed to sediment in surface runoff in HRU for day (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sedminps](#)  
*amount of stable mineral phosphorus sorbed to sediment in surface runoff in HRU for day (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::sedyld](#)  
*soil loss caused by water erosion for day in HRU (metric tons)*

- real \*8, dimension(:), allocatable [parm::sepbtm](#)  
*percolation from bottom of soil profile for the day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::strsn](#)  
*fraction of potential plant growth achieved on the day where the reduction is caused by nitrogen stress (none)*
- real \*8, dimension(:), allocatable [parm::sedorgp](#)  
*amount of organic phosphorus in surface runoff in HRU for the day (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::surfq](#)  
*surface runoff generated in HRU on the current day (mm H2O)*
- real \*8, dimension(:), allocatable [parm::strstmp](#)  
*fraction of potential plant growth achieved on the day in HRU where the reduction is caused by temperature stress (none)*
- real \*8, dimension(:), allocatable [parm::surqno3](#)  
*amount of nitrate transported in surface runoff in HRU for the day (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::hru\\_ha](#)  
*area of HRU in hectares (ha)*
- real \*8, dimension(:), allocatable [parm::hru\\_dafr](#)  
*fraction of total watershed area contained in HRU (km2/km2)*
- real \*8, dimension(:), allocatable [parm::drydep\\_no3](#)  
*atmospheric dry deposition of nitrates (kg/ha/yr)*
- real \*8, dimension(:), allocatable [parm::drydep\\_nh4](#)  
*atmospheric dry deposition of ammonia (kg/ha/yr)*
- real \*8, dimension(:), allocatable [parm::bio\\_yrms](#)  
*annual biomass (dry weight) in the HRU (metric tons/ha)*
- real \*8, dimension(:), allocatable [parm::phubase](#)  
*base zero total heat units (used when no land cover is growing) (heat units)*
- real \*8, dimension(:), allocatable [parm::hvstiadj](#)  
*optimal harvest index adjusted for water stress for current time during growing season ((kg/ha)/(kg/ha))*
- real \*8, dimension(:), allocatable [parm::laiday](#)  
*leaf area index for HRU ( $m^2/m^2$ )*
- real \*8, dimension(:), allocatable [parm::chlap](#)  
*chlorophyll-a production coefficient for pond (none)*
- real \*8, dimension(:), allocatable [parm::pnd\\_psed](#)  
*amount of mineral P attached to sediment originating from surface runoff in pond at end of day or beginnig of day (kg P)*
- real \*8, dimension(:), allocatable [parm::laimxfr](#)
- real \*8, dimension(:), allocatable [parm::seccip](#)  
*water clarity coefficient for pond (none)*
- real \*8, dimension(:), allocatable [parm::plantn](#)  
*amount of nitrogen in plant biomass (kg N/ha)*
- real \*8, dimension(:), allocatable [parm::plt\\_et](#)  
*actual ET simulated during life of plant (mm H2O)*
- real \*8, dimension(:), allocatable [parm::wet\\_psed](#)  
*amount of mineral P attached to sediment originating from surface runoff in wetland at end of day (kg P)*
- real \*8, dimension(:), allocatable [parm::bio\\_aams](#)  
*average annual biomass (dry weight) in the HRU (metric tons)*
- real \*8, dimension(:), allocatable [parm::plantp](#)  
*amount of phosphorus stored in plant biomass (kg P/ha)*
- real \*8, dimension(:), allocatable [parm::plt\\_pet](#)  
*potential ET simulated during life of plant (mm H2O)*
- real \*8, dimension(:), allocatable [parm::dormhr](#)

- time threshold used to define dormant period for plant (when daylength is within the time specified by dl from the minimum daylength for the area, the plant will go dormant) (hour)*
- real \*8, dimension(:), allocatable [parm::lai\\_yrmx](#)  
*maximum leaf area index for the year in the HRU (none)*
  - real \*8, dimension(:), allocatable [parm::lat\\_pst](#)  
*amount of pesticide in lateral flow in HRU for the day (kg pst/ha)*
  - real \*8, dimension(:), allocatable **parm::orig\_snohru**
  - real \*8, dimension(:), allocatable **parm::orig\_potvol**
  - real \*8, dimension(:), allocatable [parm::pltfr\\_n](#)  
*fraction of plant biomass that is nitrogen (none)*
  - real \*8, dimension(:), allocatable **parm::orig\_alai**
  - real \*8, dimension(:), allocatable **parm::orig\_bioms**
  - real \*8, dimension(:), allocatable [parm::pltfr\\_p](#)  
*fraction of plant biomass that is phosphorus (none)*
  - real \*8, dimension(:), allocatable **parm::orig\_phuacc**
  - real \*8, dimension(:), allocatable **parm::orig\_sumix**
  - real \*8, dimension(:), allocatable [parm::phu\\_plt](#)  
*total number of heat units to bring plant to maturity (heat units)*
  - real \*8, dimension(:), allocatable **parm::orig\_phu**
  - real \*8, dimension(:), allocatable **parm::orig\_shallst**
  - real \*8, dimension(:), allocatable **parm::orig\_deepst**
  - real \*8, dimension(:), allocatable **parm::orig\_pndvol**
  - real \*8, dimension(:), allocatable **parm::orig\_pndsed**
  - real \*8, dimension(:), allocatable **parm::orig\_pndno3**
  - real \*8, dimension(:), allocatable **parm::orig\_pndsolv**
  - real \*8, dimension(:), allocatable **parm::orig\_pndorgn**
  - real \*8, dimension(:), allocatable **parm::orig\_pndorgp**
  - real \*8, dimension(:), allocatable **parm::orig\_wetvol**
  - real \*8, dimension(:), allocatable **parm::orig\_wetsed**
  - real \*8, dimension(:), allocatable **parm::orig\_wetno3**
  - real \*8, dimension(:), allocatable **parm::orig\_wetsolv**
  - real \*8, dimension(:), allocatable **parm::orig\_wetorgn**
  - real \*8, dimension(:), allocatable **parm::orig\_wetorgp**
  - real \*8, dimension(:), allocatable **parm::orig\_solcov**
  - real \*8, dimension(:), allocatable **parm::orig\_solsw**
  - real \*8, dimension(:), allocatable **parm::orig\_potno3**
  - real \*8, dimension(:), allocatable **parm::orig\_potsed**
  - real \*8, dimension(:), allocatable [parm::wtab](#)  
*water table based on 30 day antecedent climate (precip,et) (mm)*
  - real \*8, dimension(:), allocatable [parm::shallst\\_n](#)  
*nitrate concentration in shallow aquifer converted to kg/ha (ppm NO3-N)*
  - real \*8, dimension(:), allocatable **parm::gw\_nloss**
  - real \*8, dimension(:), allocatable **parm::rchrg\_n**
  - real \*8, dimension(:), allocatable **parm::det\_san**
  - real \*8, dimension(:), allocatable **parm::det\_sil**
  - real \*8, dimension(:), allocatable **parm::det\_cla**
  - real \*8, dimension(:), allocatable **parm::det\_sag**
  - real \*8, dimension(:), allocatable **parm::det\_lag**
  - real \*8, dimension(:), allocatable [parm::afrt\\_surface](#)  
*fraction of fertilizer which is applied to top 10 mm of soil (the remaining fraction is applied to first soil layer) (none)*
  - real \*8, dimension(:), allocatable [parm::tnylida](#)  
*estimated/target nitrogen content of yield used in autofertilization (kg N/kg yield)*
  - real \*8 [parm::frt\\_surface](#)

- fraction of fertilizer which is applied to the top 10 mm of soil (the remaining fraction is applied to the first soil layer)*  
(none)
- real \*8, dimension(:), allocatable [parm::auto\\_nyr](#)  
*maximum NO3-N content allowed to be applied in one year by auto-fertilization (kg NO3-N/ha)*
- real \*8, dimension(:), allocatable [parm::auto\\_napp](#)  
*maximum NO3-N content allowed in one fertilizer application (kg NO3-N/ha)*
- real \*8, dimension(:), allocatable [parm::auto\\_nstrs](#)  
*nitrogen stress factor which triggers auto fertilization (none)*
- real \*8, dimension(:), allocatable [parm::manure\\_kg](#)  
*dry weight of manure deposited on HRU daily ((kg/ha)/day)*
- real \*8, dimension(:, :), allocatable [parm::rcn\\_mo](#)
- real \*8, dimension(:, :), allocatable [parm::rammo\\_mo](#)
- real \*8, dimension(:, :), allocatable [parm::drydep\\_no3\\_mo](#)
- real \*8, dimension(:, :), allocatable [parm::drydep\\_nh4\\_mo](#)
- real \*8, dimension(:), allocatable [parm::rcn\\_d](#)
- real \*8, dimension(:), allocatable [parm::rammo\\_d](#)
- real \*8, dimension(:), allocatable [parm::drydep\\_no3\\_d](#)
- real \*8, dimension(:), allocatable [parm::drydep\\_nh4\\_d](#)
- real \*8, dimension(:, :), allocatable [parm::yldn](#)  
*average value for yield of crop (kg/ha)*
- integer, dimension(:, :), allocatable [parm::gwati](#)
- real \*8, dimension(:, :), allocatable [parm::gwatn](#)
- real \*8, dimension(:, :), allocatable [parm::gwatl](#)
- real \*8, dimension(:, :), allocatable [parm::gwatw](#)
- real \*8, dimension(:, :), allocatable [parm::gwatd](#)
- real \*8, dimension(:, :), allocatable [parm::gwats](#)
- real \*8, dimension(:, :), allocatable [parm::gwatspcon](#)
- real \*8, dimension(:, :), allocatable [parm::psetlp](#)  
*psetlp(1,:) phosphorus settling rate for 1st season (m/day)*  
*psetlp(2,:) phosphorus settling rate for 2nd season (m/day)*
- real \*8, dimension(:, :), allocatable [parm::wgnold](#)  
*previous value of wgncur(:, :) (none)*
- real \*8, dimension(:, :), allocatable [parm::wgncur](#)  
*parameter to predict the impact of precip on other weather attributes (none)*  
*wgncur(1,:) parameter which predicts impact of precip on daily maximum air temperature*  
*wgncur(2,:) parameter which predicts impact of precip on daily minimum air temperature*  
*wgncur(3,:) parameter which predicts impact of precip on daily solar radiation*
- real \*8, dimension(:, :), allocatable [parm::wrt](#)  
*wrt(1,:) 1st shape parameter for calculation of water retention (none)*  
*wrt(2,:) 2nd shape parameter for calculation of water retention (none)*
- real \*8, dimension(:, :), allocatable [parm::pst\\_enr](#)  
*pesticide enrichment ratio (none)*
- real \*8, dimension(:, :), allocatable [parm::pst\\_surq](#)  
*amount of pesticide type lost in water surface runoff on current day in HRU (kg/ha)*
- real \*8, dimension(:, :), allocatable [parm::zdb](#)  
*division term from net pesticide equation (mm)*
- real \*8, dimension(:, :), allocatable [parm::plt\\_pst](#)  
*pesticide on plant foliage (kg/ha)*
- real \*8, dimension(:, :), allocatable [parm::psetlw](#)  
*psetlw(1,:) phosphorus settling rate for 1st season (m/day)*  
*psetlw(2,:) phosphorus settling rate for 2nd season (m/day)*
- real \*8, dimension(:, :), allocatable [parm::pst\\_sed](#)  
*pesticide loading from HRU sorbed onto sediment (kg/ha)*

- real \*8, dimension(:,:), allocatable [parm::wupnd](#)  
average daily water removal from the pond for the month for the HRU within the subbasin ( $10^4 \text{ m}^3/\text{day}$ )
- real \*8, dimension(:,:), allocatable [parm::phi](#)  
*phi(1,:) cross-sectional area of flow at bankfull depth ( $\text{m}^2$ ) phi(2,:) (none) phi(3,:) (none) phi(4,:) (none) phi(5,:) flow rate when reach is at bankfull depth ( $\text{m}^3/\text{s}$ ) phi(6,:) bottom width of main channel (m) phi(7,:) depth of water when reach is at bankfull depth (m) phi(8,:) average velocity when reach is at bankfull depth (m/s) phi(9,:) wave celerity when reach is at bankfull depth (m/s) phi(10,:) storage time constant for reach at bankfull depth (ratio of storage to discharge) (hour) phi(11,:) average velocity when reach is at 0.1 bankfull depth (low flow) (m/s) phi(12,:) wave celerity when reach is at 0.1 bankfull depth (low flow) (m/s) phi(13,:) storage time constant for reach at 0.1 bankfull depth (low flow) (ratio of storage to discharge) (hour)*
- real \*8, dimension(:,:), allocatable [parm::pcpband](#)  
precipitation for the day in band in HRU (mm H<sub>2</sub>O)
- real \*8, dimension(:,:), allocatable [parm::tavband](#)  
average temperature for the day in band in HRU (deg C)
- real \*8, dimension(:), allocatable [parm::wat\\_phi1](#)  
cross-sectional area of flow at bankfull depth ( $\text{m}^2$ )
- real \*8, dimension(:), allocatable [parm::wat\\_phi5](#)  
flow rate when reach is at bankfull depth ( $\text{m}^3/\text{s}$ )
- real \*8, dimension(:), allocatable [parm::wat\\_phi6](#)  
bottom width of main channel (m)
- real \*8, dimension(:), allocatable [parm::wat\\_phi9](#)  
depth of water when reach is at bankfull depth (m)
- real \*8, dimension(:,:), allocatable [parm::snoeb](#)  
snow water content in elevation band on current day (mm H<sub>2</sub>O)
- real \*8, dimension(:,:), allocatable [parm::wudeep](#)  
average daily water removal from the deep aquifer for the month for the HRU within the subbasin ( $10^4 \text{ m}^3/\text{day}$ )
- real \*8, dimension(:,:), allocatable [parm::wushal](#)  
average daily water removal from the shallow aquifer for the month for the HRU within the subbasin ( $10^4 \text{ m}^3/\text{day}$ )
- real \*8, dimension(:,:), allocatable [parm::bss](#)  
*bss(1,:) amount of lateral flow lagged (mm H<sub>2</sub>O)  
bss(2,:) amount of nitrate in lateral flow lagged (kg N/ha)  
bss(3,:) amount of tile flow lagged (mm)  
bss(4,:) amount of nitrate in tile flow lagged (kg N/ha)*
- real \*8, dimension(:,:), allocatable [parm::nsetlw](#)  
*nsetlw(1,:) nitrogen settling rate for 1st season (m/day)  
nsetlw(2,:) nitrogen settling rate for 2nd season (m/day)*
- real \*8, dimension(:,:), allocatable [parm::snotmpeb](#)  
temperature of snow pack in elevation band (deg C)
- real \*8, dimension(:,:), allocatable [parm::surf\\_bs](#)  
*surf\_bs(1,:) amount of surface runoff lagged over one day (mm H<sub>2</sub>O)  
surf\_bs(2,:) amount of sediment yield lagged over one day (metric tons)  
surf\_bs(3,:) amount of organic nitrogen loading lagged over one day (kg N/ha)  
surf\_bs(4,:) amount of organic phosphorus loading lagged over one day (kg P/ha)  
surf\_bs(5,:) amount of nitrate loading in surface runoff lagged over one day (kg N/ha)  
surf\_bs(6,:) amount of soluble phosphorus loading lagged over one day (kg P/ha)  
surf\_bs(7,:) amount of active mineral phosphorus loading lagged over one day (kg P/ha)  
surf\_bs(8,:) amount of stable mineral phosphorus loading lagged over one day (kg P/ha)  
surf\_bs(9,:) amount of less persistent bacteria in solution lagged over one day (# colonies/ha)  
surf\_bs(10,:) amount of persistent bacteria in solution lagged over one day (# colonies/ha)  
surf\_bs(11,:) amount of less persistent bacteria sorbed lagged over one day (# colonies/ha)  
surf\_bs(12,:) amount of persistent bacteria sorbed lagged over one day (# colonies/ha)*
- real \*8, dimension(:,:), allocatable [parm::nsetlp](#)  
*nsetlp(1,:) nitrogen settling rate for 1st season (m/day)  
nsetlp(2,:) nitrogen settling rate for 2nd season (m/day)*
- real \*8, dimension(:,:), allocatable [parm::tmxband](#)



- maximum temperature for the day in band in HRU (deg C)*
- real \*8, dimension(:,:), allocatable **parm::frad**
  - fraction of solar radiation occuring during hour in day in HRU (none)*
- real \*8, dimension(:,:), allocatable **parm::rainsub**
  - precipitation for the time step during the day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable **parm::rstpbsb**
- real \*8, dimension(:,:), allocatable **parm::orig\_snoeb**
- real \*8, dimension(:,:), allocatable **parm::orig\_pltpst**
- real \*8, dimension(:,:), allocatable **parm::terr\_p**
- real \*8, dimension(:,:), allocatable **parm::terr\_cn**
- real \*8, dimension(:,:), allocatable **parm::terr\_sl**
- real \*8, dimension(:,:), allocatable **parm::drain\_d**
- real \*8, dimension(:,:), allocatable **parm::drain\_t**
- real \*8, dimension(:,:), allocatable **parm::drain\_g**
- real \*8, dimension(:,:), allocatable **parm::drain\_idep**
- real \*8, dimension(:,:), allocatable **parm::cont\_cn**
- real \*8, dimension(:,:), allocatable **parm::cont\_p**
- real \*8, dimension(:,:), allocatable **parm::strip\_n**
- real \*8, dimension(:,:), allocatable **parm::strip\_cn**
- real \*8, dimension(:,:), allocatable **parm::strip\_p**
- real \*8, dimension(:,:), allocatable **parm::fire\_cn**
- integer, dimension(:,:), allocatable **parm::cropno\_upd**
- real \*8, dimension(:,:), allocatable **parm::hi\_upd**
- real \*8, dimension(:,:), allocatable **parm::laimx\_upd**
- real \*8, dimension(:,:), allocatable **parm::pst\_lag**
  - pst\_lag(1,:): amount of soluble pesticide in surface runoff lagged (kg pst/ha)*
  - pst\_lag(2,:): amount of sorbed pesticide in surface runoff lagged (kg pst/ha)*
  - pst\_lag(3,:): amount of pesticide lagged (kg pst/ha)*
- integer, dimension(:), allocatable **parm::hrupest**
  - pesticide use flag (none)*
  - 0: no pesticides used in HRU*
  - 1: pesticides used in HRU*
- integer, dimension(:), allocatable **parm::swtrg**
  - rainfall event flag (none):*
  - 0: no rainfall event over midnight*
  - 1: rainfall event over midnight*
- integer, dimension(:), allocatable **parm::igro**
  - land cover status code (none). This code informs the model whether or not a land cover is growing at the beginning of the simulation*
  - 0 no land cover currently growing*
  - 1 land cover growing*
- integer, dimension(:,:), allocatable **parm::ipnd**
  - ipnd(1,:) beginning month of 2nd "season" of nutrient settling season (none)*
  - ipnd(2,:) ending month of 2nd "season" of nutrient settling season (none)*
- integer, dimension(:,:), allocatable **parm::iflod**
  - iflod(1,:) beginning month of non-flood season (none)*
  - iflod(2,:) ending month of non-flood season (none)*
- integer, dimension(:), allocatable **parm::ndtarg**
  - number of days required to reach target storage from current pond storage (none)*
- integer, dimension(:), allocatable **parm::nstress**
  - code for approach used to determine amount of nitrogen to HRU (none):*
  - 0 nitrogen target approach*
  - 1 annual max approach*
- integer, dimension(:), allocatable **parm::iafrttyp**
- integer, dimension(:), allocatable **parm::igrotree**

- integer, dimension(:), allocatable [parm::grz\\_days](#)  
*number of days grazing will be simulated (none)*
- integer, dimension(:), allocatable [parm::nmgt](#)  
*management code (for GIS output only) (none)*
- integer, dimension(:), allocatable [parm::icr](#)  
*sequence number of crop grown within the current year (none)*
- integer, dimension(:), allocatable [parm::irrno](#)  
*irrigation source location (none)*  
*if IRRSC=1, IRRNO is the number of the reach*  
*if IRRSC=2, IRRNO is the number of the reservoir*  
*if IRRSC=3, IRRNO is the number of the subbasin*  
*if IRRSC=4, IRRNO is the number of the subbasin*  
*if IRRSC=5, not used*
- integer, dimension(:), allocatable [parm::sol\\_nly](#)  
*number of soil layers in HRU (none)*
- integer, dimension(:), allocatable [parm::npcp](#)  
*prior day category (none)*  
*1 dry day*  
*2 wet day*
- integer, dimension(:), allocatable [parm::irn](#)  
*average annual number of irrigation applications in HRU (none)*
- integer, dimension(:), allocatable [parm::igrz](#)  
*grazing flag for HRU (none):*  
*0 HRU currently not grazed*  
*1 HRU currently grazed*
- integer, dimension(:), allocatable [parm::ndeat](#)  
*number of days HRU has been grazed (days)*
- integer, dimension(:), allocatable [parm::hru\\_sub](#)  
*subbasin number in which HRU/reach is located (none)*
- integer, dimension(:), allocatable [parm::urbu](#)  
*urban land type identification number from urban database (urban.dat) (none)*
- integer, dimension(:), allocatable [parm::ldrain](#)  
*soil layer where drainage tile is located (none)*
- integer, dimension(:), allocatable [parm::idorm](#)  
*dormancy status code (none):*  
*0 land cover growing (not dormant)*  
*1 land cover dormant*
- integer, dimension(:), allocatable [parm::hru\\_seq](#)
- integer, dimension(:), allocatable [parm::iurban](#)  
*urban simulation code (none):*  
*0 no urban sections in HRU*  
*1 urban sections in HRU, simulate using USGS regression equations*  
*2 urban sections in HRU, simulate using build up/wash off algorithm*
- integer, dimension(:), allocatable [parm::icfrt](#)  
*continuous fertilizer flag for HRU (none):*  
*0 HRU currently not continuously fertilized*  
*1 HRU currently continuously fertilized*
- integer, dimension(:), allocatable [parm::iday\\_fert](#)
- integer, dimension(:), allocatable [parm::hrugis](#)  
*GIS code printed to output files (output.hru, output.rch) (none)*
- integer, dimension(:), allocatable [parm::ndcfrt](#)  
*number of days HRU has been continuously fertilized (days)*
- integer, dimension(:), allocatable [parm::irrsc](#)



- irrigation source code (none):*
  - 1 divert water from reach
  - 2 divert water from reservoir
  - 3 divert water from shallow aquifer
  - 4 divert water from deep aquifer
  - 5 divert water from source outside watershed
- integer, dimension(:), allocatable **parm::orig\_igro**
- integer, dimension(:), allocatable **parm::curyr\_mat**
- integer, dimension(:), allocatable **parm::icpst**
  - icpst = 0 do not apply*
  - icpst = 1 application period*
- integer, dimension(:), allocatable **parm::ndcpst**
  - current day within the application period (day)*
- integer, dimension(:), allocatable **parm::iday\_pest**
  - current day between applications (day)*
- integer, dimension(:), allocatable **parm::irr\_flag**
- integer, dimension(:,:), allocatable **parm::rndseed**
  - random number generator seeds array. The seeds in the array are used to generate random numbers for the following purposes (none):*
    - (1) wet/dry day probability
    - (2) solar radiation
    - (3) precipitation
    - (4) USLE rainfall erosion index
    - (5) wind speed
    - (6) 0.5 hr rainfall fraction
    - (7) relative humidity
    - (8) maximum temperature
    - (9) minimum temperature
    - (10) generate new random numbers
- integer, dimension(:,:), allocatable **parm::ncrops**
- integer, dimension(:), allocatable **parm::manure\_id**
  - manure (fertilizer) identification number from fert.dat (none)*
- integer, dimension(:,:), allocatable **parm::idplot**
- integer, dimension(:,:), allocatable **parm::iopday**
- integer, dimension(:,:), allocatable **parm::iopyr**
- integer, dimension(:,:), allocatable **parm::mgt\_ops**
- real \*8, dimension(:), allocatable **parm::wshd\_pstap**
  - total or average annual amount of pesticide type applied in watershed during simulation (kg/ha)*
- real \*8, dimension(:), allocatable **parm::wshd\_pstdg**
  - amount or average annual of pesticide lost through degradation in watershed (kg pst/ha)*
- integer, dimension(12) **parm::ndmo**
  - cumulative number of days accrued in the month since the simulation began where the array location number is the number of the month (days)*
- integer, dimension(:), allocatable **parm::npno**
  - array of unique pesticides used in watershed (none)*
- integer, dimension(:), allocatable **parm::mcrhru**
- character(len=13), dimension(18) **parm::rfile**
  - rainfall file names (.pcp)*
- character(len=13), dimension(18) **parm::tfile**
  - temperature file names (.tmp)*
- character(len=4), dimension(1000) **parm::urbname**
  - name of urban land use*
- character(len=1), dimension(:), allocatable **parm::hydgrp**
- character(len=16), dimension(:), allocatable **parm::snam**
  - soil series name*

- character(len=17), dimension(300) **parm::pname**  
*name of pesticide/toxin*
- character(len=4), dimension(60) **parm::title**  
*title description lines in file.cio (1st 3 lines)*
- character(len=4), dimension(5000) **parm::cpnm**  
*four character code to represent crop name*
- character(len=17), dimension(50) **parm::fname**
- real \*8, dimension(:, :, :), allocatable **parm::flomon**  
*average amount of water loaded to stream on a given day in the month ( $m^3/day$ )*
- real \*8, dimension(:, :, :), allocatable **parm::solpstmon**  
*average daily soluble pesticide loading for month (mg pst/day)*
- real \*8, dimension(:, :, :), allocatable **parm::srpstmon**  
*average daily sorbed pesticide loading for month (mg pst/day)*
- real \*8, dimension(:, :, :), allocatable **parm::orgnmon**  
*average amount of organic N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:, :, :), allocatable **parm::orgpmon**  
*average amount of organic P loaded to stream on a given day in the month (kg P/day)*
- real \*8, dimension(:, :, :), allocatable **parm::sedmon**  
*average amount of sediment loaded to stream on a given day in the month (metric tons/d)*
- real \*8, dimension(:, :, :), allocatable **parm::minpmon**  
*average amount of soluble P loaded to stream on a given day in the month (kg P/day)*
- real \*8, dimension(:, :, :), allocatable **parm::nh3mon**  
*average amount of NH3-N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:, :, :), allocatable **parm::no3mon**  
*average amount of NO3-N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:, :, :), allocatable **parm::bactlpmon**  
*average amount of less persistent bacteria loaded to stream on a given day in the month (# bact/day)*
- real \*8, dimension(:, :, :), allocatable **parm::bactpmon**  
*average amount of persistent bacteria loaded to stream on a given day in the month (# bact/day)*
- real \*8, dimension(:, :, :), allocatable **parm::no2mon**  
*average amount of NO2-N loaded to stream on a given day in the month (kg N/day)*
- real \*8, dimension(:, :, :), allocatable **parm::cmtlmon**  
*cmtlmon(1, :, :) average amount of conservative metal #1 loaded to stream on a given day in the month (# bact/day)*  
*cmtlmon(2, :, :) average amount of conservative metal #2 loaded to stream on a given day in the month (# bact/day)*  
*cmtlmon(3, :, :) average amount of conservative metal #3 loaded to stream on a given day in the month (# bact/day)*
- real \*8, dimension(:, :, :), allocatable **parm::cbodmon**  
*average amount of CBOD loaded to stream on a given day in the month (kg/day)*
- real \*8, dimension(:, :, :), allocatable **parm::chlamon**  
*average amount of chlorophyll a loaded to stream on a given day in the month (kg/day)*
- real \*8, dimension(:, :, :), allocatable **parm::disoxmon**  
*average amount of dissolved oxygen loaded to stream on a given day in the month (kg/day)*
- real \*8, dimension(:, :), allocatable **parm::floyr**  
*average daily water loading for year ( $m^3/day$ )*
- real \*8, dimension(:, :), allocatable **parm::orgnyr**  
*average daily organic N loading for year (kg N/day)*
- real \*8, dimension(:, :), allocatable **parm::orgpyr**  
*average daily organic P loading for year (kg P/day)*
- real \*8, dimension(:, :), allocatable **parm::sedyr**  
*average daily sediment loading for year (metric tons/day)*
- real \*8, dimension(:, :), allocatable **parm::minpyr**  
*average daily mineral P loading for year (kg P/day)*

- real \*8, dimension(:,:), allocatable [parm::nh3yr](#)  
*average daily NH3-N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [parm::no2yr](#)  
*average daily NO2-N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [parm::no3yr](#)  
*average daily NO3-N loading for year (kg N/day)*
- real \*8, dimension(:,:), allocatable [parm::bactlpyr](#)  
*average daily loading of less persistent bacteria for year (# bact/day)*
- real \*8, dimension(:,:), allocatable [parm::bactpyr](#)  
*average daily loading of persistent bacteria for year (# bact/day)*
- real \*8, dimension(:,:), allocatable [parm::cmtlyr](#)  
*cmtlyr(1,:,:) average daily loading of conservative metal #1 for year (kg/day)*  
*cmtlyr(2,:,:) average daily loading of conservative metal #2 for year (kg/day)*  
*cmtlyr(3,:,:) average daily loading of conservative metal #3 for year (kg/day)*
- real \*8, dimension(:,:), allocatable [parm::chlayr](#)  
*average daily loading of chlorophyll-a in year (kg/day)*
- real \*8, dimension(:,:), allocatable [parm::cbodyr](#)  
*average daily loading of CBOD in year (kg/day)*
- real \*8, dimension(:,:), allocatable [parm::disoxyr](#)  
*average daily loading of dissolved O2 in year (kg/day)*
- real \*8, dimension(:,:), allocatable [parm::solpstyr](#)  
*average daily soluble pesticide loading for year (mg pst/day)*
- real \*8, dimension(:,:), allocatable [parm::srbpstyr](#)  
*average daily sorbed pesticide loading for year (mg pst/day)*
- real \*8, dimension(:,:), allocatable [parm::sol\\_mc](#)
- real \*8, dimension(:,:), allocatable [parm::sol\\_mn](#)
- real \*8, dimension(:,:), allocatable [parm::sol\\_mp](#)
- real \*8, dimension(:), allocatable [parm::flocnst](#)  
*average daily water loading to reach ( $m^3$  H2O/day)*
- real \*8, dimension(:), allocatable [parm::orgncnst](#)  
*average daily organic N loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [parm::sedcnst](#)  
*average daily sediment loading for reach (metric tons/day)*
- real \*8, dimension(:), allocatable [parm::minpcnst](#)  
*average daily soluble P loading to reach (kg P/day)*
- real \*8, dimension(:), allocatable [parm::no3cnst](#)  
*average daily nitrate loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [parm::orgpcnst](#)  
*average daily organic P loading to reach (kg P/day)*
- real \*8, dimension(:), allocatable [parm::bactpcnst](#)  
*average daily persistent bacteria loading to reach (# bact/day)*
- real \*8, dimension(:), allocatable [parm::nh3cnst](#)  
*average daily ammonia loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [parm::no2cnst](#)  
*average daily nitrite loading to reach (kg N/day)*
- real \*8, dimension(:), allocatable [parm::bactlpcnst](#)  
*average daily less persistent bacteria loading to reach (# bact/day)*
- real \*8, dimension(:,:), allocatable [parm::cmtlcnst](#)  
*cmtlcnst(1,:) average daily conservative metal #1 loading (kg/day)*  
*cmtlcnst(2,:) average daily conservative metal #2 loading (kg/day)*  
*cmtlcnst(3,:) average daily conservative metal #3 loading (kg/day)*
- real \*8, dimension(:), allocatable [parm::chlacnst](#)

- average daily chlorophyll-a loading to reach (kg/day)*
- real \*8, dimension(:), allocatable [parm::disoxcnst](#)
  - average daily dissolved oxygen loading to reach (kg/day)*
- real \*8, dimension(:), allocatable [parm::cbodcnst](#)
  - average daily loading of CBOD to reach (kg/day)*
- real \*8, dimension(:), allocatable [parm::solpstcnst](#)
  - average daily soluble pesticide loading (mg/day)*
- real \*8, dimension(:), allocatable [parm::srbspstcnst](#)
  - average daily sorbed pesticide loading (mg/day)*
- integer [parm::nstep](#)
  - max number of time steps per day or number of lines of rainfall data for each day (depends on model operational time step) (none)*
- integer [parm::idt](#)
  - length of time step used to report precipitation data for sub-daily modeling (operational time step) (minutes)*
- real \*8, dimension(:), allocatable [parm::hdepth](#)
  - depth of flow during hour (m)*
- real \*8, dimension(:), allocatable [parm::hhstor](#)
  - water stored in reach at end of hour ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable [parm::hrtwtr](#)
  - water leaving reach in hour ( $m^3$ )*
- real \*8, dimension(:), allocatable [parm::hsdti](#)
  - flow rate in reach for hour ( $m^3/s$ )*
- real \*8, dimension(:), allocatable [parm::hrchwtr](#)
  - water stored in reach at beginning of hour ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable [parm::hnh4](#)
  - ammonia concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [parm::horgn](#)
  - organic nitrogen concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [parm::halgae](#)
- real \*8, dimension(:), allocatable [parm::hbod](#)
  - carbonaceous biochemical oxygen demand in reach at end of hour (mg O<sub>2</sub>/L)*
- real \*8, dimension(:), allocatable [parm::hno2](#)
  - nitrite concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [parm::hno3](#)
  - nitrate concentration in reach at end of hour (mg N/L)*
- real \*8, dimension(:), allocatable [parm::horgp](#)
  - organic phosphorus concentration in reach at end of hour (mg P/L)*
- real \*8, dimension(:), allocatable [parm::hsolp](#)
  - dissolved phosphorus concentration in reach at end of hour (mg P/L)*
- real \*8, dimension(:), allocatable [parm::hchla](#)
  - chlorophyll-a concentration in reach at end of hour (mg chl-a/L)*
- real \*8, dimension(:), allocatable [parm::hdisox](#)
  - dissolved oxygen concentration in reach at end of hour (mg O<sub>2</sub>/L)*
- real \*8, dimension(:), allocatable [parm::hsedyld](#)
  - sediment transported out of reach during hour (metric tons)*
- real \*8, dimension(:), allocatable [parm::hsedst](#)
- real \*8, dimension(:), allocatable [parm::hharea](#)
  - cross-sectional area of flow ( $m^2$ )*
- real \*8, dimension(:), allocatable [parm::hsolpst](#)
  - soluble pesticide concentration in outflow on day (mg pst/ $m^3$ )*
- real \*8, dimension(:), allocatable [parm::hsorpst](#)

- sorbed pesticide concentration in outflow on day (mg pst/m<sup>3</sup>)*
- real \*8, dimension(:), allocatable [parm::hhqday](#)
  - surface runoff generated each timestep of day in HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::precipdt](#)
  - precipitation, or effective precipitation reaching soil surface, in time step for HRU (mm H2O)*
- real \*8, dimension(:), allocatable [parm::hhtime](#)
  - travel time of flow in reach for hour (hour)*
- real \*8, dimension(:), allocatable [parm::hbactlp](#)
  - less persistent bacteria in reach/outflow during hour (# cfu/100mL)*
- real \*8, dimension(:), allocatable [parm::hbactp](#)
  - persistent bacteria in reach/outflow during hour (# cfu/100mL)*
- integer, dimension(6) **parm::ivar\_orig**
- real \*8, dimension(4) **parm::rvar\_orig**
- integer **parm::iatmodep**
- real \*8, dimension(:), allocatable **parm::wattemp**
- real \*8, dimension(:), allocatable **parm::lkpst\_mass**
- real \*8, dimension(:), allocatable **parm::lkspst\_mass**
- real \*8, dimension(:), allocatable **parm::vel\_chan**
- real \*8, dimension(:), allocatable [parm::vfscn](#)
  - fraction of the total runoff from the entire field entering the most concentrated 10% of the VFS (none)*
- real \*8, dimension(:), allocatable [parm::vfsratio](#)
  - field area/VFS area ratio (none)*
- real \*8, dimension(:), allocatable [parm::vfsch](#)
  - fraction of flow entering the most concentrated 10% of the VFS which is fully channelized (none)*
- real \*8, dimension(:), allocatable **parm::vfsi**
- real \*8, dimension(:,:), allocatable **parm::filter\_i**
- real \*8, dimension(:,:), allocatable **parm::filter\_ratio**
- real \*8, dimension(:,:), allocatable **parm::filter\_con**
- real \*8, dimension(:,:), allocatable **parm::filter\_ch**
- real \*8, dimension(:,:), allocatable **parm::sol\_n**
- integer [parm::cswat](#)
  - = 0 Static soil carbon (old mineralization routines)*
  - = 1 C-FARM one carbon pool model*
  - = 2 Century model*
- real \*8, dimension(:,:), allocatable **parm::tillagef**
- real \*8, dimension(:), allocatable **parm::rtfr**
- real \*8, dimension(:), allocatable [parm::stsol\\_rd](#)
  - storing last soil root depth for use in harvestkilop/kilop (mm)*
- integer **parm::dorm\_flag**
- real \*8 **parm::bf\_flg**
- real \*8 **parm::iabstr**
- real \*8, dimension(:), allocatable [parm::ubntss](#)
  - TSS loading from urban impervious cover (metric tons)*
- real \*8, dimension(:), allocatable [parm::ubnrunoff](#)
  - surface runoff from urban impervious cover (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::sub\\_ubnrunoff](#)
  - surface runoff from urban impervious cover in subbasin (mm H2O)*
- real \*8, dimension(:,:), allocatable [parm::sub\\_ubntss](#)
  - TSS loading from urban impervious cover in subbasin (metric tons)*
- real \*8, dimension(:,:,:), allocatable **parm::hhsurf\_bs**
- integer [parm::iuh](#)
  - unit hydrograph method: 1=triangular UH; 2=gamma function UH;*

- integer `parm::sed_ch`  
*channel routing for HOURLY; 0=Bagnold; 2=Brownlie; 3=Yang;*
- real \*8 `parm::eros_expo`  
*an exponent in the overland flow erosion equation ranges 1.5-3.0*
- real \*8 `parm::eros_spl`  
*coefficient of splash erosion varing 0.9-3.1*
- real \*8 `parm::rill_mult`  
*Multiplier to USLE\_K for soil susceptible to rill erosion, range 0.5-2.0.*
- real \*8 `parm::c_factor`
- real \*8 `parm::ch_d50`  
*median particle diameter of channel bed (mm)*
- real \*8 `parm::sig_g`  
*geometric standard deviation of particle sizes for the main channel. Mean air temperature at which precipitation is equally likely to be rain as snow/freezing rain.*
- real \*8 `parm::uhalpha`  
*alpha coefficient for estimating unit hydrograph using a gamma function (\*.bsn)*
- real \*8 `parm::abstinit`
- real \*8, dimension(:,:), allocatable `parm::hhsedy`  
*sediment yield from HRU drung a time step applied to HRU (tons)*
- real \*8, dimension(:,:), allocatable `parm::sub_subp_dt`  
*precipitation for time step in subbasin (mm H2O)*
- real \*8, dimension(:,:), allocatable `parm::sub_hhsedy`  
*sediment yield for the time step in subbasin (metric tons)*
- real \*8, dimension(:,:), allocatable `parm::sub_atmp`
- real \*8, dimension(:), allocatable `parm::rhy`  
*main channel hydraulic radius (m H2O)*
- real \*8, dimension(:), allocatable `parm::hrtevp`  
*evaporation losses for hour ( $m^3$  H2O)*
- real \*8, dimension(:), allocatable `parm::hrttlc`  
*transmission losses for hour ( $m^3$  H2O)*
- real \*8, dimension(:), allocatable `parm::dratio`
- real \*8, dimension(:,:), allocatable `parm::rchhr`
- real \*8, dimension(:), allocatable `parm::hhresflwo`
- real \*8, dimension(:), allocatable `parm::hhressedo`
- character(len=4), dimension(30) `parm::lu_nodrain`
- integer, dimension(:), allocatable `parm::bmpdrain`
- real \*8, dimension(:), allocatable `parm::sub_cn2`
- real \*8, dimension(:), allocatable `parm::sub_ha_urb`
- real \*8, dimension(:), allocatable `parm::bmp_recharge`
- real \*8, dimension(:), allocatable `parm::sub_ha_imp`
- real \*8, dimension(:), allocatable `parm::subdr_km`
- real \*8, dimension(:), allocatable `parm::subdr_ickm`
- real \*8, dimension(:,:), allocatable `parm::sf_im`
- real \*8, dimension(:,:), allocatable `parm::sf_iy`
- real \*8, dimension(:,:), allocatable `parm::sp_sa`
- real \*8, dimension(:,:), allocatable `parm::sp_pvol`
- real \*8, dimension(:,:), allocatable `parm::sp_pd`
- real \*8, dimension(:,:), allocatable `parm::sp_sedi`
- real \*8, dimension(:,:), allocatable `parm::sp_sede`
- real \*8, dimension(:,:), allocatable `parm::ft_sa`
- real \*8, dimension(:,:), allocatable `parm::ft_fsa`
- real \*8, dimension(:,:), allocatable `parm::ft_dep`
- real \*8, dimension(:,:), allocatable `parm::ft_h`

- real \*8, dimension(:,:), allocatable **parm::ft\_pd**
- real \*8, dimension(:,:), allocatable **parm::ft\_k**
- real \*8, dimension(:,:), allocatable **parm::ft\_dp**
- real \*8, dimension(:,:), allocatable **parm::ft\_dc**
- real \*8, dimension(:,:), allocatable **parm::ft\_por**
- real \*8, dimension(:,:), allocatable **parm::tss\_den**
- real \*8, dimension(:,:), allocatable **parm::ft\_alp**
- real \*8, dimension(:,:), allocatable **parm::sf\_fr**
- real \*8, dimension(:,:), allocatable **parm::sp\_qi**
- real \*8, dimension(:,:), allocatable **parm::sp\_k**
- real \*8, dimension(:,:), allocatable **parm::ft\_qpnd**
- real \*8, dimension(:,:), allocatable **parm::sp\_dp**
- real \*8, dimension(:,:), allocatable **parm::ft\_qsw**
- real \*8, dimension(:,:), allocatable **parm::ft\_qin**
- real \*8, dimension(:,:), allocatable **parm::ft\_qout**
- real \*8, dimension(:,:), allocatable **parm::ft\_sedpnd**
- real \*8, dimension(:,:), allocatable **parm::sp\_bpw**
- real \*8, dimension(:,:), allocatable **parm::ft\_bpw**
- integer, dimension(:), allocatable **parm::num\_sf**
- integer, dimension(:,:), allocatable **parm::sf\_typ**
- integer, dimension(:,:), allocatable **parm::sf\_dim**
- integer, dimension(:,:), allocatable **parm::ft\_qfg**
- integer, dimension(:,:), allocatable **parm::sp\_qfg**
- integer, dimension(:,:), allocatable **parm::sf\_ptp**
- real \*8, dimension(:,:), allocatable **parm::ft\_fc**
- real \*8 **parm::sfsedmean**
- real \*8 **parm::sfsedstdev**
- integer, dimension(:), allocatable **parm::dtp\_imo**  
*month the reservoir becomes operational (none)*
- integer, dimension(:), allocatable **parm::dtp\_iyr**  
*year of the simulation that the reservoir becomes operational (none)*
- integer, dimension(:), allocatable **parm::dtp\_numstage**  
*total number of stages in the weir (none)*
- integer, dimension(:), allocatable **parm::dtp\_numweir**  
*total number of weirs in the BMP (none)*
- integer, dimension(:), allocatable **parm::dtp\_onoff**  
*sub-basin detention pond is associated with (none)*
- integer, dimension(:), allocatable **parm::dtp\_relytype**  
*equations for stage-discharge relationship (none):*  
*1=exponential function,*  
*2=linear,*  
*3=logarithmic,*  
*4=cubic,*  
*5=power*
- integer, dimension(:), allocatable **parm::dtp\_stagdis**  
*(none):*  
*0=use weir/orifice discharge equation to calculate outflow,*  
*1=use stage-dicharge relationship*
- real \*8, dimension(:), allocatable **parm::cf**  
*this parameter controls the response of decomposition to the combined effect of soil temperature and moisture.*
- real \*8, dimension(:), allocatable **parm::cfh**  
*maximum humification rate*
- real \*8, dimension(:), allocatable **parm::cfdec**



*the undisturbed soil turnover rate under optimum soil water and temperature. Increasing it will increase carbon and organic N decomp.*

- real \*8, dimension(:), allocatable **parm::lat\_orgn**
- real \*8, dimension(:), allocatable **parm::lat\_orgp**
- integer, dimension(:,:), allocatable **parm::dtp\_weirdim**  
*weir dimensions (none),*  
*1=read user input,*  
*0=use model calculation*
- integer, dimension(:,:), allocatable **parm::dtp\_weirtype**  
*type of weir (none):*  
*1=rectangular and*  
*2=circular*
- real \*8, dimension(:,:), allocatable **parm::dtp\_coef**  
*dtp\_coef(1,:) coefficient of 3rd degree in the polynomial equation (none)*  
*dtp\_coef(2,:) coefficient of 2nd degree in the polynomial equation (none)*  
*dtp\_coef(3,:) coefficient of 1st degree in the polynomial equation (none)*
- real \*8, dimension(:), allocatable **parm::dtp\_evsv**  
*detention pond evaporation coefficient (none)*
- real \*8, dimension(:), allocatable **parm::dtp\_expont**  
*exponent used in the exponential equation (none)*
- real \*8, dimension(:), allocatable **parm::dtp\_intcept**  
*intercept used in regression equations (none)*
- real \*8, dimension(:), allocatable **parm::dtp\_lwratio**  
*ratio of length to width of water back up (none)*
- real \*8, dimension(:), allocatable **parm::dtp\_totwrwid**  
*total constructed width of the detention wall across the creek (m)*
- real \*8, dimension(:), allocatable **parm::dtp\_ivol**
- real \*8, dimension(:), allocatable **parm::dtp\_ised**
- integer, dimension(:,:), allocatable **parm::ro\_bmp\_flag**
- real \*8, dimension(:,:), allocatable **parm::psp\_store**
- real \*8, dimension(:,:), allocatable **parm::ssp\_store**
- real \*8, dimension(:,:), allocatable **parm::sol\_cal**
- real \*8, dimension(:,:), allocatable **parm::sol\_ph**
- integer **parm::sol\_p\_model**
- integer, dimension(:,:), allocatable **parm::a\_days**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_flo**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_sed**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_bac**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_pp**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_sp**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_pn**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_sn**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_flos**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_seds**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_bacs**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_pps**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_sps**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_pns**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_sns**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_flot**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_sedt**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_bact**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_ppt**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_spt**



- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_pnt**
- real \*8, dimension(:,:), allocatable **parm::ro\_bmp\_snt**
- real \*8, dimension(:), allocatable **parm::bmp\_flo**
- real \*8, dimension(:), allocatable **parm::bmp\_sed**
- real \*8, dimension(:), allocatable **parm::bmp\_bac**
- real \*8, dimension(:), allocatable **parm::bmp\_pp**
- real \*8, dimension(:), allocatable **parm::bmp\_sp**
- real \*8, dimension(:), allocatable **parm::bmp\_pn**
- real \*8, dimension(:), allocatable **parm::bmp\_sn**
- real \*8, dimension(:), allocatable **parm::bmp\_flag**
- real \*8, dimension(:), allocatable **parm::bmp\_flos**
- real \*8, dimension(:), allocatable **parm::bmp\_seds**
- real \*8, dimension(:), allocatable **parm::bmp\_bacs**
- real \*8, dimension(:), allocatable **parm::bmp\_pps**
- real \*8, dimension(:), allocatable **parm::bmp\_sps**
- real \*8, dimension(:), allocatable **parm::bmp\_pns**
- real \*8, dimension(:), allocatable **parm::bmp\_sns**
- real \*8, dimension(:), allocatable **parm::bmp\_flot**
- real \*8, dimension(:), allocatable **parm::bmp\_sedt**
- real \*8, dimension(:), allocatable **parm::bmp\_bact**
- real \*8, dimension(:), allocatable **parm::bmp\_ppt**
- real \*8, dimension(:), allocatable **parm::bmp\_spt**
- real \*8, dimension(:), allocatable **parm::bmp\_pnt**
- real \*8, dimension(:), allocatable **parm::bmp\_snt**
- real \*8, dimension(:,:), allocatable **parm::dtp\_addon**  
*the distance between spillway levels (m)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_cdis**  
*discharge coefficient for weir/orifice flow at different stages (none)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_depweir**  
*depth of rectangular weir at different stages (m)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_diaweir**  
*diameter of circular weir at different stages (m)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_flowrate**  
*maximum discharge from each stage of the weir/hole ( $m^3/s$ )*
- real \*8, dimension(:,:), allocatable **parm::dtp\_pcpret**  
*precipitation for different return periods (not used) (mm)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_retperd**  
*return period at different stages (years)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_wdratio**  
*width depth ratio of rectangular weirs at different stages (none)*
- real \*8, dimension(:,:), allocatable **parm::dtp\_wrwid**
- real \*8, dimension(:), allocatable **parm::ri\_subkm**
- real \*8, dimension(:), allocatable **parm::ri\_totpvol**
- real \*8, dimension(:), allocatable **parm::irmmdt**
- real \*8, dimension(:,:), allocatable **parm::ri\_sed**  
*total sediment deposited in the pond (tons)*
- real \*8, dimension(:,:), allocatable **parm::ri\_fr**
- real \*8, dimension(:,:), allocatable **parm::ri\_dim**
- real \*8, dimension(:,:), allocatable **parm::ri\_im**
- real \*8, dimension(:,:), allocatable **parm::ri\_iy**
- real \*8, dimension(:,:), allocatable **parm::ri\_sa**
- real \*8, dimension(:,:), allocatable **parm::ri\_vol**
- real \*8, dimension(:,:), allocatable **parm::ri\_qi**

- real \*8, dimension(:,:), allocatable **parm::ri\_k**
- real \*8, dimension(:,:), allocatable **parm::ri\_dd**
- real \*8, dimension(:,:), allocatable **parm::ri\_evrsv**
- real \*8, dimension(:,:), allocatable **parm::ri\_dep**
- real \*8, dimension(:,:), allocatable **parm::ri\_ndt**
- real \*8, dimension(:,:), allocatable **parm::ri\_pmpvol**
- real \*8, dimension(:,:), allocatable **parm::ri\_sed\_cumul**
- real \*8, dimension(:,:), allocatable **parm::hrnopcp**
- real \*8, dimension(:,:), allocatable **parm::ri\_qloss**
- real \*8, dimension(:,:), allocatable **parm::ri\_pumpv**
- real \*8, dimension(:,:), allocatable **parm::ri\_sedi**
- character(len=4), dimension(:,:), allocatable **parm::ri\_nirr**
- integer, dimension(:), allocatable **parm::num\_ri**
- integer, dimension(:), allocatable **parm::ri\_luflg**
- integer, dimension(:), allocatable **parm::num\_noirr**
- integer, dimension(:), allocatable **parm::wtp\_subnum**
- integer, dimension(:), allocatable **parm::wtp\_onoff**
- integer, dimension(:), allocatable **parm::wtp\_imo**
- integer, dimension(:), allocatable **parm::wtp\_iyr**
- integer, dimension(:), allocatable **parm::wtp\_dim**
- integer, dimension(:), allocatable **parm::wtp\_stagdis**
- integer, dimension(:), allocatable **parm::wtp\_sdtype**
- real \*8, dimension(:), allocatable **parm::wtp\_evrsv**  
*detention pond evaporation coefficient (none)*
- real \*8, dimension(:), allocatable **parm::wtp\_pvol**  
*volume of permanent pool including forebay ( $m^3 H_2O$ )*
- real \*8, dimension(:), allocatable **parm::wtp\_pdepth**
- real \*8, dimension(:), allocatable **parm::wtp\_sdslope**
- real \*8, dimension(:), allocatable **parm::wtp\_lenwidth**
- real \*8, dimension(:), allocatable **parm::wtp\_extdepth**
- real \*8, dimension(:), allocatable **parm::wtp\_hydeff**
- real \*8, dimension(:), allocatable **parm::wtp\_sdintc**
- real \*8, dimension(:), allocatable **parm::wtp\_sdexp**
- real \*8, dimension(:), allocatable **parm::wtp\_sdc1**
- real \*8, dimension(:), allocatable **parm::wtp\_sdc2**
- real \*8, dimension(:), allocatable **parm::wtp\_sdc3**
- real \*8, dimension(:), allocatable **parm::wtp\_pdia**
- real \*8, dimension(:), allocatable **parm::wtp\_plen**
- real \*8, dimension(:), allocatable **parm::wtp\_pmann**
- real \*8, dimension(:), allocatable **parm::wtp\_ploss**
- real \*8, dimension(:), allocatable **parm::wtp\_k**
- real \*8, dimension(:), allocatable **parm::wtp\_dp**
- real \*8, dimension(:), allocatable **parm::wtp\_sedi**
- real \*8, dimension(:), allocatable **parm::wtp\_sede**
- real \*8, dimension(:), allocatable **parm::wtp\_qi**
- real \*8 **parm::lai\_init**  
*initial leaf area index of transplants*
- real \*8 **parm::bio\_init**  
*initial biomass of transplants (kg/ha)*
- real \*8 **parm::cnop**  
*SCS runoff curve number for moisture condition II (none)*
- real \*8 **parm::harveff**  
*harvest efficiency: fraction of harvested yield that is removed from HRU; the remainder becomes residue on the soil surface(none)*

- real \*8 **parm::hi\_ovr**  
*harvest index target specified at harvest ((kg/ha)/(kg/ha))*
- real \*8 **parm::frac\_harvk**
- real \*8 **parm::lid\_vgcl**  
*van Genuchten equation's coefficient, l (none)*
- real \*8 **parm::lid\_vgcm**  
*van Genuchten equation's coefficient, m (none)*
- real \*8 **parm::lid\_qsurf\_total**
- real \*8 **parm::lid\_farea\_sum**
- real \*8, dimension(:,:), allocatable **parm::lid\_cuminf\_last**  
*cumulative amount of water infiltrated into the amended soil layer at the last time step in a day (mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::lid\_cumr\_last**  
*cumulative amount of rainfall at the last time step in a day (mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::lid\_excum\_last**  
*cumulative amount of excess rainfall at the last time step in a day (mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::lid\_f\_last**  
*potential infiltration rate of the amended soil layer at the last time step in a day (mm/mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::lid\_sw\_last**  
*soil water content of the amended soil layer at the last time step in a day (mm/mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::lid\_qsurf**  
*depth of runoff generated on a LID in a given time interval (mm H2O)*
- real \*8, dimension(:,:), allocatable **parm::interval\_last**
- real \*8, dimension(:,:), allocatable **parm::lid\_str\_last**
- real \*8, dimension(:,:), allocatable **parm::lid\_farea**
- real \*8, dimension(:,:), allocatable **parm::lid\_sw\_add**
- real \*8, dimension(:,:), allocatable **parm::lid\_cumqperc\_last**
- real \*8, dimension(:,:), allocatable **parm::lid\_cumirr\_last**
- integer, dimension(:,:), allocatable **parm::gr\_onoff**
- integer, dimension(:,:), allocatable **parm::gr\_imo**
- integer, dimension(:,:), allocatable **parm::gr\_iyr**
- real \*8, dimension(:,:), allocatable **parm::gr\_farea**  
*fractional area of a green roof to the HRU (none)*
- real \*8, dimension(:,:), allocatable **parm::gr\_solop**
- real \*8, dimension(:,:), allocatable **parm::gr\_etcoef**
- real \*8, dimension(:,:), allocatable **parm::gr\_fc**
- real \*8, dimension(:,:), allocatable **parm::gr\_wp**
- real \*8, dimension(:,:), allocatable **parm::gr\_ksat**
- real \*8, dimension(:,:), allocatable **parm::gr\_por**
- real \*8, dimension(:,:), allocatable **parm::gr\_hydeff**
- real \*8, dimension(:,:), allocatable **parm::gr\_soldpt**
- integer, dimension(:,:), allocatable **parm::rg\_onoff**
- integer, dimension(:,:), allocatable **parm::rg\_imo**
- integer, dimension(:,:), allocatable **parm::rg\_iyr**
- real \*8, dimension(:,:), allocatable **parm::rg\_farea**
- real \*8, dimension(:,:), allocatable **parm::rg\_solop**
- real \*8, dimension(:,:), allocatable **parm::rg\_etcoef**
- real \*8, dimension(:,:), allocatable **parm::rg\_fc**
- real \*8, dimension(:,:), allocatable **parm::rg\_wp**
- real \*8, dimension(:,:), allocatable **parm::rg\_ksat**
- real \*8, dimension(:,:), allocatable **parm::rg\_por**
- real \*8, dimension(:,:), allocatable **parm::rg\_hydeff**
- real \*8, dimension(:,:), allocatable **parm::rg\_soldpt**
- real \*8, dimension(:,:), allocatable **parm::rg\_dimop**

- real \*8, dimension(:,:), allocatable **parm::rg\_sarea**
- real \*8, dimension(:,:), allocatable **parm::rg\_vol**
- real \*8, dimension(:,:), allocatable **parm::rg\_sth**
- real \*8, dimension(:,:), allocatable **parm::rg\_sdia**
- real \*8, dimension(:,:), allocatable **parm::rg\_bdia**
- real \*8, dimension(:,:), allocatable **parm::rg\_sts**
- real \*8, dimension(:,:), allocatable **parm::rg\_orifice**
- real \*8, dimension(:,:), allocatable **parm::rg\_oheight**
- real \*8, dimension(:,:), allocatable **parm::rg\_odia**
- integer, dimension(:,:), allocatable **parm::cs\_onoff**
- integer, dimension(:,:), allocatable **parm::cs\_imo**
- integer, dimension(:,:), allocatable **parm::cs\_iyr**
- integer, dimension(:,:), allocatable **parm::cs\_grcon**
- real \*8, dimension(:,:), allocatable **parm::cs\_farea**
- real \*8, dimension(:,:), allocatable **parm::cs\_vol**
- real \*8, dimension(:,:), allocatable **parm::cs\_rdepth**
- integer, dimension(:,:), allocatable **parm::pv\_onoff**
- integer, dimension(:,:), allocatable **parm::pv\_imo**
- integer, dimension(:,:), allocatable **parm::pv\_iyr**
- integer, dimension(:,:), allocatable **parm::pv\_solop**
- real \*8, dimension(:,:), allocatable **parm::pv\_grvdep**
- real \*8, dimension(:,:), allocatable **parm::pv\_grvpor**
- real \*8, dimension(:,:), allocatable **parm::pv\_farea**
- real \*8, dimension(:,:), allocatable **parm::pv\_drcoef**
- real \*8, dimension(:,:), allocatable **parm::pv\_fc**
- real \*8, dimension(:,:), allocatable **parm::pv\_wp**
- real \*8, dimension(:,:), allocatable **parm::pv\_ksat**
- real \*8, dimension(:,:), allocatable **parm::pv\_por**
- real \*8, dimension(:,:), allocatable **parm::pv\_hydeff**
- real \*8, dimension(:,:), allocatable **parm::pv\_soldpt**
- integer, dimension(:,:), allocatable **parm::lid\_onoff**
- real \*8, dimension(:,:), allocatable **parm::sol\_hsc**  
*mass of C present in slow humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_hsn**  
*mass of N present in slow humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_hpc**  
*mass of C present in passive humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_hpn**  
*mass of N present in passive humus (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_lm**  
*mass of metabolic litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_lmc**  
*mass of C in metabolic litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_lmn**  
*mass of N in metabolic litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_ls**  
*mass of structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_lsc**  
*mass of C in structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_lsl**  
*mass of lignin in structural litter (kg ha-1)*
- real \*8, dimension(:,:), allocatable **parm::sol\_lsn**  
*mass of N in structural litter (kg ha-1)*

- real \*8, dimension(:,:), allocatable **parm::sol\_bmc**
- real \*8, dimension(:,:), allocatable **parm::sol\_bmn**
- real \*8, dimension(:,:), allocatable **parm::sol\_rnmn**
- real \*8, dimension(:,:), allocatable **parm::sol\_lslc**
- real \*8, dimension(:,:), allocatable **parm::sol\_lslnc**
- real \*8, dimension(:,:), allocatable **parm::sol\_rspc**
- real \*8, dimension(:,:), allocatable **parm::sol\_woc**
- real \*8, dimension(:,:), allocatable **parm::sol\_won**
- real \*8, dimension(:,:), allocatable **parm::sol\_hp**
- real \*8, dimension(:,:), allocatable **parm::sol\_hs**
- real \*8, dimension(:,:), allocatable **parm::sol\_bm**
- real \*8, dimension(:,:), allocatable **parm::sol\_cac**
- real \*8, dimension(:,:), allocatable **parm::sol\_cec**
- real \*8, dimension(:,:), allocatable **parm::sol\_percc**
- real \*8, dimension(:,:), allocatable **parm::sol\_latc**
- real \*8, dimension(:), allocatable **parm::sedc\_d**  
*amount of C lost with sediment pools (kg C/ha)*
- real \*8, dimension(:), allocatable **parm::surfqc\_d**
- real \*8, dimension(:), allocatable **parm::latc\_d**
- real \*8, dimension(:), allocatable **parm::percc\_d**
- real \*8, dimension(:), allocatable **parm::foc\_d**
- real \*8, dimension(:), allocatable **parm::nppc\_d**
- real \*8, dimension(:), allocatable **parm::rsdc\_d**
- real \*8, dimension(:), allocatable **parm::grainc\_d**
- real \*8, dimension(:), allocatable **parm::stoverc\_d**
- real \*8, dimension(:), allocatable **parm::soc\_d**
- real \*8, dimension(:), allocatable **parm::rspc\_d**
- real \*8, dimension(:), allocatable **parm::emitc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_sedc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_surfqc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_latc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_percc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_foc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_nppc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_rsdc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_grainc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_stoverc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_emitc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_soc\_d**
- real \*8, dimension(:), allocatable **parm::sub\_rspc\_d**
- real \*8, dimension(:), allocatable **parm::sedc\_m**
- real \*8, dimension(:), allocatable **parm::surfqc\_m**
- real \*8, dimension(:), allocatable **parm::latc\_m**
- real \*8, dimension(:), allocatable **parm::percc\_m**
- real \*8, dimension(:), allocatable **parm::foc\_m**
- real \*8, dimension(:), allocatable **parm::nppc\_m**
- real \*8, dimension(:), allocatable **parm::rsdc\_m**
- real \*8, dimension(:), allocatable **parm::grainc\_m**
- real \*8, dimension(:), allocatable **parm::stoverc\_m**
- real \*8, dimension(:), allocatable **parm::emitc\_m**
- real \*8, dimension(:), allocatable **parm::soc\_m**
- real \*8, dimension(:), allocatable **parm::rspc\_m**
- real \*8, dimension(:), allocatable **parm::sedc\_a**
- real \*8, dimension(:), allocatable **parm::surfqc\_a**

- real \*8, dimension(:), allocatable **parm::latc\_a**
- real \*8, dimension(:), allocatable **parm::percc\_a**
- real \*8, dimension(:), allocatable **parm::foc\_a**
- real \*8, dimension(:), allocatable **parm::nppc\_a**
- real \*8, dimension(:), allocatable **parm::rsdc\_a**
- real \*8, dimension(:), allocatable **parm::grainc\_a**
- real \*8, dimension(:), allocatable **parm::stoverc\_a**
- real \*8, dimension(:), allocatable **parm::emitc\_a**
- real \*8, dimension(:), allocatable **parm::soc\_a**
- real \*8, dimension(:), allocatable **parm::rspc\_a**
- integer, dimension(:), allocatable **parm::tillage\_switch**
- real \*8, dimension(:), allocatable **parm::tillage\_depth**
- integer, dimension(:), allocatable **parm::tillage\_days**
- real \*8, dimension(:), allocatable **parm::tillage\_factor**
- real \*8 **parm::dthy**  
*time interval for subdaily flood routing*
- integer, dimension(4) **parm::ihx**
- integer, dimension(:), allocatable **parm::nhy**
- real \*8, dimension(:), allocatable **parm::rchx**
- real \*8, dimension(:), allocatable **parm::rcss**
- real \*8, dimension(:), allocatable **parm::qcap**
- real \*8, dimension(:), allocatable **parm::chxa**
- real \*8, dimension(:), allocatable **parm::chxp**
- real \*8, dimension(:, :, :), allocatable **parm::qhy**
- real \*8 **parm::ff1**
- real \*8 **parm::ff2**

### 5.103.1 Detailed Description

file containing the module parm

Author

modified by Javier Burguete Tolosa

## 5.104 ndenit.f90 File Reference

### Functions/Subroutines

- subroutine **ndenit** (k, j, cdg, wdn, void)  
*this subroutine computes denitrification*

### 5.104.1 Detailed Description

file containing the subroutine ndenit

Author

modified by Javier Burguete

## 5.104.2 Function/Subroutine Documentation

### 5.104.2.1 ndenit()

```
subroutine ndenit (
    integer, intent(in) k,
    integer, intent(in) j,
    real*8, intent(in) cdg,
    real*8, intent(out) wdn,
    real*8, intent(in) void )
```

this subroutine computes denitrification

#### Parameters

in	<i>k</i>	
in	<i>j</i>	
in	<i>cdg</i>	
in	<i>wdn</i>	
out	<i>void</i>	

## 5.105 newtillmix.f90 File Reference

### Functions/Subroutines

- subroutine [newtillmix](#) (j, bmix)

*this subroutine mixes residue and nutrients during tillage and biological mixing. Mixing was extended to all layers. A subroutine to simulate stimulation of organic matter decomposition was added. March 2009: testing has been minimal and further adjustments are expected. Use with caution!*

### 5.105.1 Detailed Description

file containing the subroutine newtillmix

#### Author

Armen R. Kemanian,  
Stefan Julich,  
Cole Rossi  
modified by Javier Burguete

### 5.105.2 Function/Subroutine Documentation

### 5.105.2.1 newtillmix()

```
subroutine newtillmix (
    integer, intent(in) j,
    real*8, intent(in) bmix )
```

this subroutine mixes residue and nutrients during tillage and biological mixing. Mixing was extended to all layers. A subroutine to simulate stimulation of organic matter decomposition was added. March 2009: testing has been minimal and further adjustments are expected. Use with caution!

#### Parameters

in	<i>j</i>	HRU number (none)
in	<i>bmix</i>	biological mixing efficiency: this number is zero for tillage operations (none)

## 5.106 nfix.f90 File Reference

### Functions/Subroutines

- subroutine [nfix](#) (j)  
*this subroutine estimates nitrogen fixation by legumes*

### 5.106.1 Detailed Description

file containing the subroutine nfix

#### Author

modified by Javier Burguete

### 5.106.2 Function/Subroutine Documentation

#### 5.106.2.1 nfix()

```
subroutine nfix (
    integer, intent(in) j )
```

this subroutine estimates nitrogen fixation by legumes

#### Parameters

in	<i>j</i>	HRU number
----	----------	------------



## 5.107 nitvol.f90 File Reference

### Functions/Subroutines

- subroutine `nitvol` (*j*)

*this subroutine estimates daily mineralization (NH<sub>3</sub> to NO<sub>3</sub>) and volatilization of NH<sub>3</sub>*

#### 5.107.1 Detailed Description

file containing the subroutine `nitvol`

##### Author

modified by Javier Burguete

#### 5.107.2 Function/Subroutine Documentation

##### 5.107.2.1 `nitvol()`

```
subroutine nitvol (
    integer, intent(in) j )
```

this subroutine estimates daily mineralization (NH<sub>3</sub> to NO<sub>3</sub>) and volatilization of NH<sub>3</sub>

##### Parameters

<i>j</i>	HRU number
----------	------------

## 5.108 nlch.f90 File Reference

### Functions/Subroutines

- subroutine `nlch` (*j*)

*this subroutine simulates the loss of nitrate via surface runoff, lateral flow, tile flow, and percolation out of the profile*

#### 5.108.1 Detailed Description

file containing the subroutine `nlch`

##### Author

modified by Javier Burguete

## 5.108.2 Function/Subroutine Documentation

### 5.108.2.1 nlch()

```
subroutine nlch (
    integer, intent(in) j )
```

this subroutine simulates the loss of nitrate via surface runoff, lateral flow, tile flow, and percolation out of the profile

#### Parameters

in	j	HRU number
----	---	------------

## 5.109 nminrl.f90 File Reference

### Functions/Subroutines

- subroutine [nminrl](#) (j)

*this subroutine estimates daily nitrogen and phosphorus mineralization and immobilization considering fresh organic material (plant residue) and active and stable humus material*

### 5.109.1 Detailed Description

file containing the subroutine nminrl

#### Author

modified by Javier Burguete

## 5.109.2 Function/Subroutine Documentation

### 5.109.2.1 nminrl()

```
subroutine nminrl (
    integer, intent(in) j )
```

this subroutine estimates daily nitrogen and phosphorus mineralization and immobilization considering fresh organic material (plant residue) and active and stable humus material

## Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.110 noqual.f90 File Reference

### Functions/Subroutines

- subroutine [noqual](#) (jrch, k)

*this subroutine performs in-stream nutrient calculations. No transformations are calculated. New concentrations of the nutrients are calculated based on the loading to the reach from upstream.*

#### 5.110.1 Detailed Description

file containing the subroutine noqual

## Author

modified by Javier Burguete

#### 5.110.2 Function/Subroutine Documentation

##### 5.110.2.1 noqual()

```
subroutine noqual (
    integer, intent(in) jrch,
    integer, intent(in) k )
```

this subroutine performs in-stream nutrient calculations. No transformations are calculated. New concentrations of the nutrients are calculated based on the loading to the reach from upstream.

## Parameters

in	<i>jrch</i>	reach number (none)
in	<i>k</i>	inflow hydrograph storage location number (none)

## 5.111 npup.f90 File Reference

### Functions/Subroutines

- subroutine [npup](#) (j)

*this subroutine calculates plant phosphorus uptake*

### 5.111.1 Detailed Description

file containing the subroutine npup

Author

modified by Javier Burguete

### 5.111.2 Function/Subroutine Documentation

#### 5.111.2.1 npup()

```
subroutine npup (
    integer, intent(in) j )
```

this subroutine calculates plant phosphorus uptake

Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.112 nrain.f90 File Reference

### Functions/Subroutines

- subroutine [nrain](#) (j)  
*this subroutine adds nitrate from rainfall to the soil profile*

#### 5.112.1 Detailed Description

file containing the subroutine nrain

Author

modified by Javier Burguete

### 5.112.2 Function/Subroutine Documentation

#### 5.112.2.1 nrain()

```
subroutine nrain (
    integer, intent(in) j )
```

this subroutine adds nitrate from rainfall to the soil profile

## Parameters

in	j	HRU number
----	---	------------

## 5.113 nup.f90 File Reference

### Functions/Subroutines

- subroutine `nup` (j)  
*this subroutine calculates plant nitrogen uptake*

#### 5.113.1 Detailed Description

file containing the subroutine `nup`

## Author

modified by Javier Burguete

#### 5.113.2 Function/Subroutine Documentation

##### 5.113.2.1 `nup()`

```
subroutine nup (
    integer, intent(in) j )
```

this subroutine calculates plant nitrogen uptake

## Parameters

in	j	HRU number
----	---	------------

## 5.114 nuts.f90 File Reference

### Functions/Subroutines

- subroutine `nuts` (u1, u2, uu)  
*this function calculates the plant stress factor caused by limited supply of nitrogen or phosphorus*

### 5.114.1 Detailed Description

file containing the subroutine nuts

Author

modified by Javier Burguete

### 5.114.2 Function/Subroutine Documentation

#### 5.114.2.1 nuts()

```
subroutine nuts (
    real*8, intent(in) u1,
    real*8, intent(in) u2,
    real*8, intent(out) uu )
```

this function calculates the plant stress factor caused by limited supply of nitrogen or phosphorus

Parameters

in	<i>u1</i>	actual amount of element in plant (kg/ha)
in	<i>u2</i>	optimal amount of element in plant (kg/ha)
out	<i>uu</i>	fraction of optimal plant growth achieved where reduction is caused by plant element deficiency (none)

## 5.115 openwth.f90 File Reference

### Functions/Subroutines

- subroutine [openwth](#)

*this subroutine opens the precipitation, temperature, solar radiation, relative humidity and wind speed files for simulations using measured weather data*

#### 5.115.1 Detailed Description

file containing the subroutine openwth

Author

modified by Javier Burguete

## 5.116 operatn.f90 File Reference

### Functions/Subroutines

- subroutine `operatn` (*j*)  
*this subroutine performs all management operations*

#### 5.116.1 Detailed Description

file containing the subroutine `operatn`

##### Author

modified by Javier Burguete

#### 5.116.2 Function/Subroutine Documentation

##### 5.116.2.1 `operatn()`

```
subroutine operatn (
    integer, intent(in) j )
```

this subroutine performs all management operations

##### Parameters

<i>in</i>	<i>j</i>	HRU number
-----------	----------	------------

## 5.117 orgn.f90 File Reference

### Functions/Subroutines

- subroutine `orgn` (*iwave*, *j*)  
*this subroutine calculates the amount of organic nitrogen removed in surface runoff*

#### 5.117.1 Detailed Description

file containing the subroutine `orgn`

##### Author

modified by Javier Burguete

## 5.117.2 Function/Subroutine Documentation

### 5.117.2.1 orgn()

```
subroutine orgn (
    integer, intent(in) iwave,
    integer, intent(in) j )
```

this subroutine calculates the amount of organic nitrogen removed in surface runoff

#### Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number

## 5.118 orgncswat.f90 File Reference

### Functions/Subroutines

- subroutine [orgncswat](#) (iwave, j)

*this subroutine calculates the amount of organic nitrogen removed in surface runoff - when using CSWAT it excludes sol\_aorgn, uses only sol\_n = sol\_orgn, and includes sol\_mn (nitrogen in manure)*

### 5.118.1 Detailed Description

file containing the subroutine orgncswat

#### Author

modified by Javier Burguete

## 5.118.2 Function/Subroutine Documentation

### 5.118.2.1 orgncswat()

```
subroutine orgncswat (
    integer, intent(in) iwave,
    integer, intent(in) j )
```

this subroutine calculates the amount of organic nitrogen removed in surface runoff - when using CSWAT it excludes sol\_aorgn, uses only sol\_n = sol\_orgn, and includes sol\_mn (nitrogen in manure)



## Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number

## 5.119 orgncswat2.f90 File Reference

### Functions/Subroutines

- subroutine [orgncswat2](#) (iwave, j)

*this subroutine calculates the amount of organic nitrogen removed in surface runoff - when using CSWAT==2 it*

#### 5.119.1 Detailed Description

file containing the subroutine orgncswat2

#### Author

modified by Javier Burguete

#### 5.119.2 Function/Subroutine Documentation

##### 5.119.2.1 orgncswat2()

```
subroutine orgncswat2 (
    integer, intent(in) iwave,
    integer, intent(in) j )
```

this subroutine calculates the amount of organic nitrogen removed in surface runoff - when using CSWAT==2 it

## Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number

## 5.120 origtile.f90 File Reference

### Functions/Subroutines

- subroutine [origtile](#) (d, j)

*this subroutine computes tile drainage using basic tile equations developed by Saleh et al.(2005)*

### 5.120.1 Detailed Description

file containing the subroutine origtile

#### Author

modified by Javier Burguete

### 5.120.2 Function/Subroutine Documentation

#### 5.120.2.1 origtile()

```
subroutine origtile (
    real*8, intent(in) d,
    integer, intent(in) j )
```

this subroutine computes tile drainage using basic tile equations developed by Saleh et al.(2005)

#### Parameters

in	<i>d</i>	
in	<i>j</i>	HRU number

## 5.121 ovr\_sed.f90 File Reference

### Functions/Subroutines

- subroutine [ovr\\_sed](#) (j, sb)

*this subroutine computes splash erosion by raindrop impact and flow erosion by overland flow*

### 5.121.1 Detailed Description

file containing the subroutine ovr\_sed

#### Author

modified by Javier Burguete

## 5.121.2 Function/Subroutine Documentation

### 5.121.2.1 ovr\_sed()

```
subroutine ovr_sed (
    integer, intent(in) j,
    integer, intent(in) sb )
```

this subroutine computes splash erosion by raindrop impact and flow erosion by overland flow

#### Parameters

in	<i>j</i>	HRU number (none)
in	<i>sb</i>	subbasin number (none)

## 5.122 oxygen\_saturation.f90 File Reference

### Functions/Subroutines

- real \*8 function [oxygen\\_saturation](#) (t)  
*this function calculates saturation concentration for dissolved oxygen QUAL2E section 3.6.1 equation III-29*

### 5.122.1 Detailed Description

file containing the function oxygen\_saturation

#### Author

modified by Javier Burguete

## 5.122.2 Function/Subroutine Documentation

### 5.122.2.1 oxygen\_saturation()

```
real*8 function oxygen_saturation (
    real*8, intent(in) t )
```

this function calculates saturation concentration for dissolved oxygen QUAL2E section 3.6.1 equation III-29

**Parameters**

in	<i>t</i>	temperature (deg C)
----	----------	---------------------

**Returns**

saturation concentration for dissolved oxygen

## 5.123 percmacro.f90 File Reference

### Functions/Subroutines

- subroutine [percmacro](#) (*j*)  
*this subroutine computes percolation by crack flow*

#### 5.123.1 Detailed Description

file containing the subroutine percmacro

**Author**

modified by Javier Burguete

#### 5.123.2 Function/Subroutine Documentation

##### 5.123.2.1 percmacro()

```
subroutine percmacro (
    integer, intent(in) j )
```

this subroutine computes percolation by crack flow

**Parameters**

in	<i>j</i>	HRU number
----	----------	------------

## 5.124 percmmain.f90 File Reference

### Functions/Subroutines

- subroutine [percmmain](#) (*j*, *sb*)  
*this subroutine is the master soil percolation component*

### 5.124.1 Detailed Description

file containing the subroutine percmmain

#### Author

modified by Javier Burguete

### 5.124.2 Function/Subroutine Documentation

#### 5.124.2.1 percmmain()

```
subroutine percmmain (  
    integer, intent(in) j,  
    integer, intent(in) sb )
```

this subroutine is the master soil percolation component

#### Parameters

in	j	HRU number
in	sb	subbasin number

## 5.125 percmicro.f90 File Reference

### Functions/Subroutines

- subroutine [percmicro](#) (ly1, j)

*this subroutine computes percolation and lateral subsurface flow from a soil layer when field capacity is exceeded*

### 5.125.1 Detailed Description

file containing the subroutine percmicro

#### Author

modified by Javier Burguete

### 5.125.2 Function/Subroutine Documentation

**5.125.2.1 percmicro()**

```
subroutine percmicro (
    integer, intent(in) ly1,
    integer, intent(in) j )
```

this subroutine computes percolation and lateral subsurface flow from a soil layer when field capacity is exceeded

## Parameters

in	<i>ly1</i>	soil layer number
in	<i>j</i>	HRU number

## 5.126 pestlch.f90 File Reference

### Functions/Subroutines

- subroutine [pestlch](#) (*j*)  
*this subroutine calculates pesticides leached through each layer, pesticide transported with lateral subsurface flow, and pesticide transported with surface runoff*

#### 5.126.1 Detailed Description

file containing the subroutine pestlch

#### Author

modified by Javier Burguete

#### 5.126.2 Function/Subroutine Documentation

##### 5.126.2.1 pestlch()

```
subroutine pestlch (
    integer, intent(in) j )
```

this subroutine calculates pesticides leached through each layer, pesticide transported with lateral subsurface flow, and pesticide transported with surface runoff

## Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.127 pestw.f90 File Reference

### Functions/Subroutines

- subroutine [pestw](#)  
*this suroutine writes summary information on pesticide fate in watershed*

### 5.127.1 Detailed Description

file containing the subroutine pestw

Author

modified by Javier Burguete

## 5.128 pesty.f90 File Reference

### Functions/Subroutines

- subroutine [pesty](#) (iwave, j)

### 5.128.1 Detailed Description

file containing the subroutine pesty

Author

modified by Javier Burguete

### 5.128.2 Function/Subroutine Documentation

#### 5.128.2.1 pesty()

```
subroutine pesty (  
    integer, intent(in) iwave,  
    integer, intent(in) j )
```

Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number

## 5.129 pgen.f90 File Reference

### Functions/Subroutines

- subroutine [pgen](#) (j)



*this subroutine generates precipitation data when the user chooses to simulate or when data is missing for particular days in the weather file*

### 5.129.1 Detailed Description

file containing the subroutine pgen

Author

modified by Javier Burguete

### 5.129.2 Function/Subroutine Documentation

#### 5.129.2.1 pgen()

```
subroutine pgen (
    integer, intent(in) j )
```

this subroutine generates precipitation data when the user chooses to simulate or when data is missing for particular days in the weather file

Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.130 pgenhr.f90 File Reference

### Functions/Subroutines

- subroutine [pgenhr](#) (jj)

*this subroutine distributes daily rainfall exponentially within the day @parameter[in] jj HRU number*

### 5.130.1 Detailed Description

file containing the subroutine pgenhr

Author

modified by Javier Burguete

## 5.131 pipeflow.f90 File Reference

### Functions/Subroutines

- real \*8 function [pipeflow](#) (d, h)

*this function calculates orifice pipe flow and returns flow rate ( $m^3/s$ )*

#### 5.131.1 Detailed Description

file containing the function pipeflow

#### Author

modified by Javier Burguete

#### 5.131.2 Function/Subroutine Documentation

##### 5.131.2.1 pipeflow()

```
real*8 function pipeflow (
    real*8, intent(in) d,
    real*8, intent(in) h )
```

this function calculates orifice pipe flow and returns flow rate ( $m^3/s$ )

#### Parameters

in	$d$	diameter (mm)
in	$h$	depth (mm)

#### Returns

flow rate ( $m^3/s$ )

## 5.132 pkq.f90 File Reference

### Functions/Subroutines

- subroutine [pkq](#) (iwave, j)

*this subroutine computes the peak runoff rate for each HRU and the entire subbasin using a modification of the rational formula*

### 5.132.1 Detailed Description

file containing the subroutine pkq

Author

modified by Javier Burguete

### 5.132.2 Function/Subroutine Documentation

#### 5.132.2.1 pkq()

```
subroutine pkq (
    integer, intent(in) iwave,
    integer, intent(in) j )
```

this subroutine computes the peak runoff rate for each HRU and the entire subbasin using a modification of the rational formula

Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU MUSLE(sedyld) each hru is calculated independently using hru area and adjusted channel length iwave = 1 subbasin # for subbasin MUSLE is computed for entire subbasin using hru weighted KLSCP
in	<i>j</i>	HRU number (none)

## 5.133 plantmod.f90 File Reference

### Functions/Subroutines

- subroutine [plantmod](#) (j)

*this subroutine predicts daily potential growth of total plant biomass and roots and calculates leaf area index. Incorporates residue for tillage functions and decays residue on ground surface. Adjusts daily dry matter based on water stress.*

### 5.133.1 Detailed Description

file containing the subroutine plantmod

Author

modified by Javier Burguete

## 5.133.2 Function/Subroutine Documentation

### 5.133.2.1 plantmod()

```
subroutine plantmod (
    integer, intent(in) j )
```

this subroutine predicts daily potential growth of total plant biomass and roots and calculates leaf area index. Incorporates residue for tillage functions and decays residue on ground surface. Adjusts daily dry matter based on water stress.

#### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.134 plantop.f90 File Reference

### Functions/Subroutines

- subroutine [plantop](#) (*j*)  
*this subroutine performs the plant operation*

### 5.134.1 Detailed Description

file containing the subroutine plantop

#### Author

modified by Javier Burguete

## 5.134.2 Function/Subroutine Documentation

### 5.134.2.1 plantop()

```
subroutine plantop (
    integer, intent(in) j )
```

this subroutine performs the plant operation

## Parameters

<i>i</i> <sub>n</sub>	<i>j</i>	HRU number
-----------------------	----------	------------

## 5.135 pmeas.f90 File Reference

### Functions/Subroutines

- subroutine [pmeas](#) (*i*)  
*this subroutine reads in precipitation data and assigns it to the proper subbasins*

#### 5.135.1 Detailed Description

file containing the subroutine pmeas

## Author

modified by Javier Burquete

#### 5.135.2 Function/Subroutine Documentation

##### 5.135.2.1 pmeas()

```
subroutine pmeas (
    integer, intent(in) i )
```

this subroutine reads in precipitation data and assigns it to the proper subbasins

## Parameters

<i>i</i> <sub>n</sub>	<i>i</i>	current day of simulation (julian date)
-----------------------	----------	---

## 5.136 pminrl.f90 File Reference

### Functions/Subroutines

- subroutine [pminrl](#) (*j*)  
*this subroutine computes p flux between the labile, active mineral and stable mineral p pools.*

### 5.136.1 Detailed Description

file containing the subroutine pminrl

Author

modified by Javier Burguete

### 5.136.2 Function/Subroutine Documentation

#### 5.136.2.1 pminrl()

```
subroutine pminrl (
    integer, intent(in) j )
```

this subroutine computes p flux between the labile, active mineral and stable mineral p pools.

Parameters

<i>j</i>	HRU number
----------	------------

## 5.137 pminrl2.f90 File Reference

### Functions/Subroutines

- subroutine [pminrl2](#) (*j*)  
*this subroutine computes p flux between the labile, active mineral and stable mineral p pools. this is the alternate phosphorus model described in [5]*

#### 5.137.1 Detailed Description

file containing the subroutine pminrl2

Author

modified by Javier Burguete

#### 5.137.2 Function/Subroutine Documentation

##### 5.137.2.1 pminrl2()

```
subroutine pminrl2 (
    integer, intent(in) j )
```

this subroutine computes p flux between the labile, active mineral and stable mineral p pools. this is the alternate phosphorus model described in [\[5\]](#)

## Parameters

<i>j</i>	HRU number
----------	------------

## 5.138 pond.f90 File Reference

### Functions/Subroutines

- subroutine [pond](#) (*k*)  
*this subroutine routes water and sediment through ponds and computes evaporation and seepage from the ponds*

#### 5.138.1 Detailed Description

file containing the subroutine `pond`

## Author

modified by Javier Burguete

#### 5.138.2 Function/Subroutine Documentation

##### 5.138.2.1 `pond()`

```
subroutine pond (
    integer, intent(in) k )
```

this subroutine routes water and sediment through ponds and computes evaporation and seepage from the ponds

## Parameters

<i>in</i>	<i>k</i>	HRU or reach number (none)
-----------	----------	----------------------------

## 5.139 pondhr.f90 File Reference

### Functions/Subroutines

- subroutine [pondhr](#) (*j*, *k*)

### 5.139.1 Detailed Description

file containing the subroutine pondhr

Author

modified by Javier Burguete

### 5.139.2 Function/Subroutine Documentation

#### 5.139.2.1 pondhr()

```
subroutine pondhr (
    integer, intent(in) j,
    integer, intent(in) k )
```

Parameters

in	<i>j</i>	HRU or reach number (none)
in	<i>k</i>	current time step of the day (none)

## 5.140 pothole.f90 File Reference

### Functions/Subroutines

- subroutine [pothole](#) (i, j)  
*this subroutine simulates depressional areas that do not drain to the stream network (potholes) and impounded areas such as rice paddies*

#### 5.140.1 Detailed Description

file containing the subroutine pothole

Author

modified by Javier Burguete

#### 5.140.2 Function/Subroutine Documentation



### 5.140.2.1 pothole()

```
subroutine pothole (  
    integer, intent(in) i,  
    integer, intent(in) j )
```

this subroutine simulates depressional areas that do not drain to the stream network (potholes) and impounded areas such as rice paddies

**Parameters**

in	<i>i</i>	current day in simulation–loop counter (none)
in	<i>j</i>	HRU number (none)

**5.141 print\_hyd.f90 File Reference****Functions/Subroutines**

- subroutine [print\\_hyd](#) (*i*)  
*this subroutine summarizes data for subbasins with multiple HRUs and*

**5.141.1 Detailed Description**

file containing the subroutine `print_hyd`

**Author**

modified by Javier Burguete

**5.141.2 Function/Subroutine Documentation****5.141.2.1 print\_hyd()**

```
subroutine print_hyd (
    integer, intent(in) i )
```

this subroutine summarizes data for subbasins with multiple HRUs and

**Parameters**

in	<i>i</i>	current day in simulation–loop counter (julian date)
----	----------	--

**5.142 psed.f90 File Reference****Functions/Subroutines**

- subroutine [psed](#) (*iwave*, *j*, *sb*)

### 5.142.1 Detailed Description

file containing the subroutine psed

Author

modified by Javier Burguete

### 5.142.2 Function/Subroutine Documentation

#### 5.142.2.1 psed()

```
subroutine psed (
    integer, intent(in) iwave,
    integer, intent(in) j,
    integer, intent(in) sb )
```

Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number
in	<i>sb</i>	subbasin number

## 5.143 qman.f90 File Reference

### Functions/Subroutines

- real \*8 function [qman](#) (x1, x2, x3, x4)

*this subroutine calculates flow rate or flow velocity using Manning's equation. If x1 is set to 1, the velocity is calculated. If x1 is set to cross-sectional area of flow, the flow rate is calculated.*

#### 5.143.1 Detailed Description

file containing the function qman

Author

modified by Javier Burguete

#### 5.143.2 Function/Subroutine Documentation

### 5.143.2.1 qman()

```
real*8 function qman (
    real*8, intent(in) x1,
    real*8, intent(in) x2,
    real*8, intent(in) x3,
    real*8, intent(in) x4 )
```

this subroutine calculates flow rate or flow velocity using Manning's equation. If x1 is set to 1, the velocity is calculated. If x1 is set to cross-sectional area of flow, the flow rate is calculated.

#### Parameters

in	x1	cross-sectional flow area or 1 (m <sup>2</sup> or none)
in	x2	hydraulic radius (m)
in	x3	Manning's "n" value for channel (none)
in	x4	average slope of channel (m/m)

#### Returns

flow rate or flow velocity (m<sup>3</sup>/s or m/s)

## 5.144 rchaa.f90 File Reference

### Functions/Subroutines

- subroutine [rchaa](#) (years)  
*this subroutine writes the average annual reach output to the .rch file*

### 5.144.1 Detailed Description

file containing the subroutine rchaa

#### Author

modified by Javier Burguete

### 5.144.2 Function/Subroutine Documentation

#### 5.144.2.1 rchaa()

```
subroutine rchaa (
    real*8, intent(in) years )
```

this subroutine writes the average annual reach output to the .rch file

## Parameters

in	years	length of simulation (years)
----	-------	------------------------------

## 5.145 rchday.f90 File Reference

### Functions/Subroutines

- subroutine [rchday](#)  
*this subroutine writes the daily reach output to the .rch file*

#### 5.145.1 Detailed Description

file containing the subroutine rchday

#### Author

modified by Javier Burguete

## 5.146 rchinit.f90 File Reference

### Functions/Subroutines

- subroutine [rchinit](#) (jrch)  
*this subroutine initializes variables for the daily simulation of the channel routing command loop*

#### 5.146.1 Detailed Description

file containing the subroutine rchinit

#### Author

modified by Javier Burguete

#### 5.146.2 Function/Subroutine Documentation

##### 5.146.2.1 rchinit()

```
subroutine rchinit (
    integer, intent(in) jrch )
```

this subroutine initializes variables for the daily simulation of the channel routing command loop

## Parameters

in	<i>jrch</i>	reach number
----	-------------	--------------

## 5.147 rchmon.f90 File Reference

### Functions/Subroutines

- subroutine [rchmon](#) (mdays)  
*this subroutine writes the monthly reach output to the .rch file*

#### 5.147.1 Detailed Description

file containing the subroutine rchmon

#### Author

modified by Javier Burguete

#### 5.147.2 Function/Subroutine Documentation

##### 5.147.2.1 rchmon()

```
subroutine rchmon (
    integer, intent(in) mdays )
```

this subroutine writes the monthly reach output to the .rch file

## Parameters

in	<i>mdays</i>	number of days simulated in month
----	--------------	-----------------------------------

## 5.148 rchuse.f90 File Reference

### Functions/Subroutines

- subroutine [rchuse](#) (jrch)  
*this subroutine removes water from reach for consumptive water use*

### 5.148.1 Detailed Description

file containing the subroutine rchuse

Author

modified by Javier Burguete

### 5.148.2 Function/Subroutine Documentation

#### 5.148.2.1 rchuse()

```
subroutine rchuse (
    integer, intent(in) jrch )
```

this subroutine removes water from reach for consumptive water use

Parameters

in	jrch	reach number (none)
----	------	---------------------

## 5.149 rchyr.f90 File Reference

### Functions/Subroutines

- subroutine [rchyr](#) (idlast)  
*this subroutine writes the annual reach output to the .rch file*

#### 5.149.1 Detailed Description

file containing the subroutine rchyr

Author

modified by Javier Burguete

#### 5.149.2 Function/Subroutine Documentation

##### 5.149.2.1 rchyr()

```
subroutine rchyr (
    integer, intent(in) idlast )
```

this subroutine writes the annual reach output to the .rch file

## Parameters

in	idlast	number of days simulated in month (none)
----	--------	--

## 5.150 readatmodep.f90 File Reference

### Functions/Subroutines

- subroutine [readatmodep](#)  
*this subroutine reads the atmospheric deposition values*

#### 5.150.1 Detailed Description

file containing the subroutine readatmodep

#### Author

modified by Javier Burguete

## 5.151 readbsn.f90 File Reference

### Functions/Subroutines

- subroutine [readbsn](#)  
*this subroutine reads data from the basin input file (.bsn). This file contains information related to processes modeled or defined at the watershed level*

#### 5.151.1 Detailed Description

file containing the suborutine readbsn

#### Author

modified by Javier Burguete

## 5.152 readchm.f90 File Reference

### Functions/Subroutines

- subroutine [readchm](#) (l)  
*This subroutine reads data from the HRU/subbasin soil chemical input file (.chm). This file contains initial amounts of pesticides/nutrients in the first soil layer. (Specifics about the first soil layer are given in the .sol file.) All data in the .chm file is optional input.*



### 5.152.1 Detailed Description

file containing the subroutine readchm

Author

modified by Javier Burguete

### 5.152.2 Function/Subroutine Documentation

#### 5.152.2.1 readchm()

```
subroutine readchm (
    integer, intent(in) l )
```

This subroutine reads data from the HRU/subbasin soil chemical input file (.chm). This file contains initial amounts of pesticides/nutrients in the first soil layer. (Specifics about the first soil layer are given in the .sol file.) All data in the .chm file is optional input.

Parameters

in	/	HRU number (none)
----	---	-------------------

## 5.153 readcnst.f90 File Reference

### Functions/Subroutines

- subroutine [readcnst](#) (jj)  
*reads in the loading information for the reccnst command*

### 5.153.1 Detailed Description

file containing the subroutine [readcnst.f90](#)

Author

modified by Javier Burguete

### 5.153.2 Function/Subroutine Documentation

#### 5.153.2.1 readcnst()

```
subroutine readcnst (
    integer, intent(in) jj )
```

reads in the loading information for the reccnst command

## Parameters

in	jj	file number associated with recnst command (none)
----	----	---

## 5.154 readfcst.f90 File Reference

### Functions/Subroutines

- subroutine [readfcst](#)  
*this subroutine reads the HRU forecast weather generator parameters from the .cst file*

#### 5.154.1 Detailed Description

file containing the subroutine readfcst

## Author

modified by Javier Burguete

## 5.155 readfert.f90 File Reference

### Functions/Subroutines

- subroutine [readfert](#)  
*this subroutine reads input parameters from the fertilizer/manure (i.e. nutrient) database (fert.dat)*

#### 5.155.1 Detailed Description

file containing the subroutine readfert

## Author

modified by Javier Burguete

## 5.156 readfig.f90 File Reference

### Functions/Subroutines

- subroutine [readfig](#)  
*reads in the routing information from the watershed configuration input file (.fig) and calculates the number of sub-basins, reaches, and reservoirs*

### 5.156.1 Detailed Description

file containing the subroutine readfig

Author

modified by Javier Burguete

## 5.157 readfile.f90 File Reference

### Functions/Subroutines

- subroutine [readfile](#)

*this subroutine opens the main input and output files and reads watershed information from the file.cio*

### 5.157.1 Detailed Description

file containing the subroutine readfile

Author

modified by Javier Burguete

## 5.158 readgw.f90 File Reference

### Functions/Subroutines

- subroutine [readgw](#) (i, j)

*this subroutine reads the parameters from the HRU/subbasin groundwater input file (.gw)*

### 5.158.1 Detailed Description

file containing the subroutine readgw

Author

modified by Javier Burguete

### 5.158.2 Function/Subroutine Documentation

#### 5.158.2.1 readgw()

```
subroutine readgw (  
    integer, intent(in) i,  
    integer, intent(in) j )
```

this subroutine reads the parameters from the HRU/subbasin groundwater input file (.gw)

**Parameters**

in	<i>i</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)

**5.159 readhru.f90 File Reference****Functions/Subroutines**

- subroutine [readhru](#) (*i*, *j*)

*this subroutine reads data from the HRU general input file (.hru). This file contains data related to general processes modeled at the HRU level.*

**5.159.1 Detailed Description**

file containing the subroutine readhru

**Author**

modified by Javier Burguete

**5.159.2 Function/Subroutine Documentation****5.159.2.1 readhru()**

```
subroutine readhru (
    integer, intent(in) i,
    integer, intent(in) j )
```

this subroutine reads data from the HRU general input file (.hru). This file contains data related to general processes modeled at the HRU level.

**Parameters**

in	<i>i</i>	subbasin number (none)
in	<i>j</i>	HRU number (none)

**5.160 readinpt.f90 File Reference****Functions/Subroutines**

- subroutine [readinpt](#)

*this subroutine calls subroutines which read input data for the databases and the HRUs*

### 5.160.1 Detailed Description

file containing the subroutine readinpt

Author

modified by Javier Burguete

## 5.161 readlup.f90 File Reference

### Functions/Subroutines

- subroutine [readlup](#)

*this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.*

### 5.161.1 Detailed Description

file containing the subroutine readlup

Author

modified by Javier Burguete

## 5.162 readlwq.f90 File Reference

### Functions/Subroutines

- subroutine [readlwq](#) (ii)

*this subroutine reads data from the lake water quality input file (.lwq). This file contains data related to initial pesticide and nutrient levels in the lake/reservoir and transformation processes occurring within the lake/reservoir. Data in the lake water quality input file is assumed to apply to all reservoirs in the watershed.*

### 5.162.1 Detailed Description

file containing the subroutine readlwq

Author

modified by Javier Burguete

### 5.162.2 Function/Subroutine Documentation

#### 5.162.2.1 readlwq()

```
subroutine readlwq (
    integer, intent(in) ii )
```

this subroutine reads data from the lake water quality input file (.lwq). This file contains data related to initial pesticide and nutrient levels in the lake/reservoir and transformation processes occurring within the lake/reservoir. Data in the lake water quality input file is assumed to apply to all reservoirs in the watershed.

## Parameters

in	ii	reservoir number (none)
----	----	-------------------------

## 5.163 readmgt.f90 File Reference

### Functions/Subroutines

- subroutine [readmgt](#) (k)

*this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.*

#### 5.163.1 Detailed Description

file containing the subroutine readmgt

## Author

modified by Javier Burguete

#### 5.163.2 Function/Subroutine Documentation

##### 5.163.2.1 readmgt()

```
subroutine readmgt (
    integer, intent(in) k )
```

this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.

## Parameters

in	k	HRU number (none)
----	---	-------------------

## 5.164 readmon.f90 File Reference

### Functions/Subroutines

- subroutine [readmon](#) (i)

*reads in the input data for the recmon command*

### 5.164.1 Detailed Description

file containing the subroutine readmon

Author

modified by Javier Burguete

## 5.165 readops.f90 File Reference

### Functions/Subroutines

- subroutine [readops](#) (k)

*this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.*

### 5.165.1 Detailed Description

file containing the subroutine readops

Author

modified by Javier Burguete

### 5.165.2 Function/Subroutine Documentation

#### 5.165.2.1 readops()

```
subroutine readops (
    integer, intent(in) k )
```

this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.

Parameters

in	k	HRU number (none)
----	---	-------------------

## 5.166 readpest.f90 File Reference

### Functions/Subroutines

- subroutine [readpest](#)

*this subroutine reads parameters from the toxin/pesticide database (pest.dat)*

### 5.166.1 Detailed Description

file containing the subroutine readpest

Author

modified by Javier Burguete

## 5.167 readplant.f90 File Reference

### Functions/Subroutines

- subroutine [readplant](#)

*this subroutine reads input parameters from the landuse/landcover database (plant.dat)*

### 5.167.1 Detailed Description

file containing the subroutine readplant

Author

modified by Javier Burguete

## 5.168 readpnd.f90 File Reference

### Functions/Subroutines

- subroutine [readpnd](#) (i)

*This subroutine reads data from the HRU/subbasin pond input file (.pnd). This file contains data related to ponds and wetlands in the HRUs/subbasins.*

### 5.168.1 Detailed Description

file containing the subroutine readpnd

Author

modified by Javier Burguete

### 5.168.2 Function/Subroutine Documentation

#### 5.168.2.1 readpnd()

```
subroutine readpnd (
    integer, intent(in) i )
```

This subroutine reads data from the HRU/subbasin pond input file (.pnd). This file contains data related to ponds and wetlands in the HRUs/subbasins.



## Parameters

in	i	subbasin number (none)
----	---	------------------------

## 5.169 readres.f90 File Reference

### Functions/Subroutines

- subroutine [readres](#) (i)  
*the purpose of this subroutine is to read in data from the reservoir input file (.res)*

#### 5.169.1 Detailed Description

file containing the subroutine readres

#### Author

modified by Javier Burguete

#### 5.169.2 Function/Subroutine Documentation

##### 5.169.2.1 readres()

```
subroutine readres (
    integer, intent(in) i )
```

the purpose of this subroutine is to read in data from the reservoir input file (.res)

## Parameters

in	i	reservoir number (none)
----	---	-------------------------

## 5.170 readrte.f90 File Reference

### Functions/Subroutines

- subroutine [readrte](#)  
*this subroutine reads data from the reach (main channel) input file (.rte). This file contains data related to channel attributes. Only one reach file should be made for each subbasin. If multiple HRUs are modeled within a subbasin, the same .rte file should be listed for all HRUs in file.cio*

### 5.170.1 Detailed Description

file containing the subroutine readrte

Author

modified by Javier Burguete

## 5.171 readru.f90 File Reference

### Functions/Subroutines

- subroutine [readru](#) (i)  
*this subroutine reads data from the sub input file (.sub). This file contains data related to routing*

### 5.171.1 Detailed Description

file containing the subroutine readru

Author

modified by Javier Burguete

### 5.171.2 Function/Subroutine Documentation

#### 5.171.2.1 readru()

```
subroutine readru (
    integer, intent(in) i )
```

this subroutine reads data from the sub input file (.sub). This file contains data related to routing

Parameters

in	i	subbasin number
----	---	-----------------

## 5.172 readsdr.f90 File Reference

### Functions/Subroutines

- subroutine [readsdr](#) (j)  
*this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.*

### 5.172.1 Detailed Description

file containing the subroutine readsdr

Author

modified by Javier Burguete

### 5.172.2 Function/Subroutine Documentation

#### 5.172.2.1 readsdr()

```
subroutine readsdr (  
    integer j )
```

this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.173 readsepticbz.f90 File Reference

### Functions/Subroutines

- subroutine [readsepticbz](#) (j)

*this subroutine reads data from the septic input file (.sep). This file contains information related to septic tanks modeled or defined at the watershed level*

### 5.173.1 Detailed Description

file containing the subroutine readsepticbz

Author

modified by Javier Burguete

### 5.173.2 Function/Subroutine Documentation

### 5.173.2.1 readsepticbz()

```
subroutine readsepticbz (  
    integer, intent(in) j )
```

this subroutine reads data from the septic input file (.sep). This file contains information related to septic tanks modeled or defined at the watershed level

## Parameters

<i>i</i> n	<i>j</i>	HRU number (none)
------------	----------	-------------------

## 5.174 readseptwq.f90 File Reference

### Functions/Subroutines

- subroutine [readseptwq](#)

*this subroutine reads input parameters from the sept wq database (septwq.dat). Information is used when a hru has septic tank.*

#### 5.174.1 Detailed Description

file containing the subroutine readseptwq

#### Author

C. Santhi, modified by Javier Burguete

#### 5.174.2 Function/Subroutine Documentation

##### 5.174.2.1 readseptwq()

```
subroutine readseptwq ( )
```

this subroutine reads input parameters from the sept wq database (septwq.dat). Information is used when a hru has septic tank.

This routine was developed by C. Santhi. Inputs for this routine are provided in septwq.dat of septic documentation. Data were compiled from [\[4\]](#) and [\[3\]](#).

## 5.175 readsno.f90 File Reference

### Functions/Subroutines

- subroutine [readsno](#) (*i*)

*this subroutine reads snow data from the HRU/subbasin soil chemical input*

### 5.175.1 Detailed Description

file containing the subroutine readsno

Author

modified by Javier Burguete

### 5.175.2 Function/Subroutine Documentation

#### 5.175.2.1 readsno()

```
subroutine readsno (
    integer, intent(in) i )
```

this subroutine reads snow data from the HRU/subbasin soil chemical input

Parameters

in	<i>i</i>	subbasin number (none)
----	----------	------------------------

## 5.176 readsol.f90 File Reference

### Functions/Subroutines

- subroutine [readsol](#) (k)

*this subroutine reads data from the HRU/subbasin soil properties file (.sol). This file contains data related to soil physical properties and general chemical properties*

#### 5.176.1 Detailed Description

file containing the subroutine readsol

Author

modified by Javier Burguete

#### 5.176.2 Function/Subroutine Documentation

##### 5.176.2.1 readsol()

```
subroutine readsol (
    integer, intent(in) k )
```

this subroutine reads data from the HRU/subbasin soil properties file (.sol). This file contains data related to soil physical properties and general chemical properties

## Parameters

in	k	HRU number
----	---	------------

## 5.177 readsub.f90 File Reference

### Functions/Subroutines

- subroutine [readsub](#) (i)

*this subroutine reads data from the HRU/subbasin general input file (.sub). This file contains data related to general processes modeled at the HRU/subbasin level.*

#### 5.177.1 Detailed Description

file containing the subroutine readsub

## Author

modified by Javier Burguete

#### 5.177.2 Function/Subroutine Documentation

##### 5.177.2.1 readsub()

```
subroutine readsub (
    integer, intent(in) i )
```

this subroutine reads data from the HRU/subbasin general input file (.sub). This file contains data related to general processes modeled at the HRU/subbasin level.

## Parameters

in	i	subbasin number (none)
----	---	------------------------

## 5.178 readswq.f90 File Reference

### Functions/Subroutines

- subroutine [readswq](#)

*this subroutine reads parameters from the subbasin instream water quality file (.swq) and initializes the QUAL2E variables which apply to the individual subbasins*

### 5.178.1 Detailed Description

file containing the subroutine readswq

Author

modified by Javier Burguete

## 5.179 readtill.f90 File Reference

### Functions/Subroutines

- subroutine [readtill](#)  
*this subroutine reads input data from tillage database (till.dat)*

### 5.179.1 Detailed Description

file containing the subroutine readtill

Author

modified by Javier Burguete

## 5.180 readurban.f90 File Reference

### Functions/Subroutines

- subroutine [readurban](#)  
*this subroutine reads input parameters from the urban database (urban.dat). Information from this database is used only if the urban buildup/washoff routines are selected for the modeling of urban areas*

### 5.180.1 Detailed Description

file containing the subroutine readurban

Author

modified by Javier Burguete

## 5.181 readwgn.f90 File Reference

### Functions/Subroutines

- subroutine [readwgn](#) (ii)  
*this subroutine reads the HRU weather generator parameters from the .wgn file*



### 5.181.1 Detailed Description

file containing the subroutine readwgn

Author

modified by Javier Burguete

### 5.181.2 Function/Subroutine Documentation

#### 5.181.2.1 readwgn()

```
subroutine readwgn (
    integer, intent(in) ii )
```

this subroutine reads the HRU weather generator parameters from the .wgn file

Parameters

in	ii	subbasin number (none)
----	----	------------------------

## 5.182 readwus.f90 File Reference

### Functions/Subroutines

- subroutine [readwus](#) (i)

*This subroutine reads data from the HRU/subbasin water use input file (.wus). The water use file extracts water from the subbasin and it is considered to be lost from the watershed. These variables should be used to remove water transported outside the watershed.*

### 5.182.1 Detailed Description

file containing the subroutine readwus

Author

modified by Javier Burguete

### 5.182.2 Function/Subroutine Documentation

### 5.182.2.1 readwus()

```
subroutine readwus (
    integer, intent(in) i )
```

This subroutine reads data from the HRU/subbasin water use input file (.wus). The water use file extracts water from the subbasin and it is considered to be lost from the watershed. These variables should be used to remove water transported outside the watershed.

#### Parameters

in	i	subbasin number
----	---	-----------------

## 5.183 readwwq.f90 File Reference

### Functions/Subroutines

- subroutine [readwwq](#)  
*this subroutine reads the watershed stream water quality input data (.wwq file) and initializes the QUAL2E variables which apply to the entire watershed*

### 5.183.1 Detailed Description

file containing the subroutine readwwq

#### Author

modified by Javier Burguete

## 5.184 readyr.f90 File Reference

### Functions/Subroutines

- subroutine [readyr](#) (i)  
*reads in the input data for the recyear command*

### 5.184.1 Detailed Description

file containing the subroutine readyr

#### Author

modified by Javier Burguete

### 5.184.2 Function/Subroutine Documentation

#### 5.184.2.1 readyr()

```
subroutine readyr (
    integer, intent(in) i )
```

reads in the input data for the recyear command

## Parameters

in	i	reservoir number (none)
----	---	-------------------------

## 5.185 reccnst.f90 File Reference

### Functions/Subroutines

- subroutine [reccnst](#) (k)

*this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are averaged over the entire period of record*

### 5.185.1 Detailed Description

file containing the subroutine reccnst

#### Author

modified by Javier Burguete

### 5.185.2 Function/Subroutine Documentation

#### 5.185.2.1 reccnst()

```
subroutine reccnst (
    integer, intent(in) k )
```

this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are averaged over the entire period of record

## Parameters

in	k	file number (none)
----	---	--------------------

## 5.186 recday.f90 File Reference

### Functions/Subroutines

- subroutine [recday](#) (k)

*this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a daily basis*

### 5.186.1 Detailed Description

file containing the subroutine recday

Author

modified by Javier Burguete

### 5.186.2 Function/Subroutine Documentation

#### 5.186.2.1 recday()

```
subroutine recday (
    integer, intent(in) k )
```

this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a daily basis

Parameters

in	k	reach number or file number (none)
----	---	------------------------------------

## 5.187 rechour.f90 File Reference

### Functions/Subroutines

- subroutine [rechour](#) (k)

*this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a hourly basis*

#### 5.187.1 Detailed Description

file containing the subroutine rechour

Author

modified by Javier Burguete

#### 5.187.2 Function/Subroutine Documentation

### 5.187.2.1 rechour()

```
subroutine rechour (
    integer, intent(in) k )
```

this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a hourly basis

## Parameters

in	k	reach number or file number (none)
----	---	------------------------------------

## 5.188 recmon.f90 File Reference

### Functions/Subroutines

- subroutine [recmon](#) (k)

*this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a monthly basis*

#### 5.188.1 Detailed Description

file containing the subroutine recmon

#### Author

modified by Javier Burguete

#### 5.188.2 Function/Subroutine Documentation

##### 5.188.2.1 recmon()

```
subroutine recmon (
    integer, intent(in) k )
```

this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on a monthly basis

## Parameters

in	k	file number (none)
----	---	--------------------

## 5.189 recyear.f90 File Reference

### Functions/Subroutines

- subroutine [recyear](#) (k)

*this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on an annual basis*

### 5.189.1 Detailed Description

file containing the subroutine recyear

Author

modified by Javier Burguete

### 5.189.2 Function/Subroutine Documentation

#### 5.189.2.1 recyear()

```
subroutine recyear (
    integer, intent(in) k )
```

this subroutine inputs measured loadings to the stream network for routing through the watershed where the records are summarized on an annual basis

Parameters

in	k	file number (none)
----	---	--------------------

## 5.190 regres.f90 File Reference

### Functions/Subroutines

- real \*8 function [regres](#) (k, j)  
*this function calculates constituent loadings to the main channel using USGS regression equations*

#### 5.190.1 Detailed Description

file containing the function regres

Author

modified by Javier Burguete

#### 5.190.2 Function/Subroutine Documentation

##### 5.190.2.1 regres()

```
real*8 function regres (
    integer, intent(in) k,
    integer, intent(in) j )
```

this function calculates constituent loadings to the main channel using USGS regression equations

**Parameters**

in	<i>k</i>	identification code for regression data (none) 1 carbonaceous oxygen demand 2 suspended solid load 3 total nitrogen 4 total phosphorus
in	<i>j</i>	HRU number (none)

**Returns**

amount of constituent removed in surface runoff (kg)

## 5.191 res.f90 File Reference

**Functions/Subroutines**

- subroutine [res](#) (*jres*)

*this subroutine routes water and sediment through reservoirs computes evaporation and seepage from the reservoir.*

### 5.191.1 Detailed Description

file containing the subroutine res

**Author**

modified by Javier Burguete

### 5.191.2 Function/Subroutine Documentation

#### 5.191.2.1 res()

```
subroutine res (  
    integer, intent(in) jres )
```

this subroutine routes water and sediment through reservoirs computes evaporation and seepage from the reservoir.

**Parameters**

in	<i>jres</i>	reservoir number (none)
----	-------------	-------------------------



## 5.192 resetlu.f90 File Reference

### Functions/Subroutines

- subroutine [resetlu](#)

*this subroutine reads data from the HRU/subbasin management input file (.mgt). This file contains data related to management practices used in the HRU/subbasin.*

### 5.192.1 Detailed Description

file containing the subroutine resetlu

Author

modified by Javier Burguete

## 5.193 reshr.f90 File Reference

### Functions/Subroutines

- subroutine [reshr](#) (jres)

*this subroutine routes water and sediment through reservoirs computes evaporation and seepage from the reservoir.*

### 5.193.1 Detailed Description

file containing the subroutine reshr

Author

modified by Javier Burguete

### 5.193.2 Function/Subroutine Documentation

#### 5.193.2.1 reshr()

```
subroutine reshr (
    integer, intent(in) jres )
```

this subroutine routes water and sediment through reservoirs computes evaporation and seepage from the reservoir.

Parameters

in	jres	reservoir number (none)
----	------	-------------------------

## 5.194 resinit.f90 File Reference

### Functions/Subroutines

- subroutine [resinit](#) (jres)

*this subroutine initializes variables for the daily simulation of the channel routing command loop*

#### 5.194.1 Detailed Description

file containing the subroutine resinit

##### Author

modified by Javier Burguete

#### 5.194.2 Function/Subroutine Documentation

##### 5.194.2.1 resinit()

```
subroutine resinit (
    integer, intent(in) jres )
```

this subroutine initializes variables for the daily simulation of the channel routing command loop

##### Parameters

in	<i>jres</i>	reservoir number
----	-------------	------------------

## 5.195 resnut.f90 File Reference

### Functions/Subroutines

- subroutine [resnut](#) (jres)

*this subroutine routes soluble nitrogen and soluble phosphorus through reservoirs*

#### 5.195.1 Detailed Description

file containing the subroutine resnut

##### Author

modified by Javier Burguete

## 5.195.2 Function/Subroutine Documentation

### 5.195.2.1 resnut()

```
subroutine resnut (
    integer, intent(in) jres )
```

this subroutine routes soluble nitrogen and soluble phosphorus through reservoirs

#### Parameters

in	jres	reservoir number (none)
----	------	-------------------------

## 5.196 rewind\_init.f90 File Reference

### Functions/Subroutines

- subroutine [rewind\\_init](#)  
*this subroutine reinitializes values for running different scenarios*

### 5.196.1 Detailed Description

file containing the subroutine rewind\_init

#### Author

modified by Javier Burguete

## 5.197 rhgen.f90 File Reference

### Functions/Subroutines

- subroutine [rhgen](#) (j)  
*this subroutine generates weather relative humidity, solar radiation, and wind speed.*

### 5.197.1 Detailed Description

file containing the subroutine rhgen

#### Author

modified by Javier Burguete

## 5.198 rootfr.f90 File Reference

### Functions/Subroutines

- subroutine `rootfr` (*j*)  
*this subroutine distributes dead root mass through the soil profile*

#### 5.198.1 Detailed Description

file containing the subroutine `rootfr`

##### Author

Armen R. Kemanian,  
modified by Javier Burguete

#### 5.198.2 Function/Subroutine Documentation

##### 5.198.2.1 `rootfr()`

```
subroutine rootfr (
    integer, intent(in) j )
```

this subroutine distributes dead root mass through the soil profile

##### Parameters

<code>in</code>	<code>j</code>	HRU number
-----------------	----------------	------------

## 5.199 route.f90 File Reference

### Functions/Subroutines

- subroutine `route` (*i*, *jrch*, *k*)  
*this subroutine simulates channel routing*

#### 5.199.1 Detailed Description

file containing the subroutine `route`

##### Author

modified by Javier Burguete

## 5.199.2 Function/Subroutine Documentation

### 5.199.2.1 route()

```
subroutine route (
    integer, intent(in) i,
    integer, intent(in) jrch,
    integer, intent(in) k )
```

this subroutine simulates channel routing

#### Parameters

in	<i>i</i>	current day in simulation—loop counter (julian date)
in	<i>jrch</i>	reach number (none)
in	<i>k</i>	inflow hydrograph storage location number (none)

## 5.200 routels.f90 File Reference

### Functions/Subroutines

- subroutine **routels** (iru\_sub, sb)

### 5.200.1 Detailed Description

file containing the subroutine routels

#### Author

modified by Javier Burguete

## 5.201 routeunit.f90 File Reference

### Functions/Subroutines

- subroutine **routeunit** (j)

### 5.201.1 Detailed Description

file containing the subroutine routeunit

#### Author

modified by Javier Burguete

## 5.201.2 Function/Subroutine Documentation

### 5.201.2.1 routeunit()

```
subroutine routeunit (
    integer, intent(in) j )
```

#### Parameters

in	<i>j</i>	reach number (none)
----	----------	---------------------

## 5.202 routres.f90 File Reference

### Functions/Subroutines

- subroutine [routres](#) (*jres*)  
*this subroutine performs reservoir routing*

### 5.202.1 Detailed Description

file containing the subroutine routres

#### Author

modified by Javier Burguete

## 5.202.2 Function/Subroutine Documentation

### 5.202.2.1 routres()

```
subroutine routres (
    integer, intent(in) jres )
```

this subroutine performs reservoir routing

#### Parameters

in	<i>jres</i>	reservoir number (none)
----	-------------	-------------------------

## 5.203 rsedaa.f90 File Reference

### Functions/Subroutines

- subroutine `rsedaa` (years)  
*this subroutine writes the annual reach output to the .sed file*

#### 5.203.1 Detailed Description

file containing the subroutine `rsedaa`

##### Author

modified by Javier Burguete

#### 5.203.2 Function/Subroutine Documentation

##### 5.203.2.1 `rsedaa()`

```
subroutine rsedaa (
    real*8, intent(in) years )
```

this subroutine writes the annual reach output to the .sed file

##### Parameters

<code>years</code>	length of simulation (years)
--------------------	------------------------------

## 5.204 rseday.f90 File Reference

### Functions/Subroutines

- subroutine `rseday`

#### 5.204.1 Detailed Description

file containing the subroutine `rseday`

##### Author

modified by Javier Burguete

## 5.205 rsedmon.f90 File Reference

### Functions/Subroutines

- subroutine `rsedmon` (mdays)  
*this subroutine writes the monthly reach output to the .sed file*

#### 5.205.1 Detailed Description

file containing the subroutine rsedmon

##### Author

modified by Javier Burguete

#### 5.205.2 Function/Subroutine Documentation

##### 5.205.2.1 rsedmon()

```
subroutine rsedmon (
    integer, intent(in) mdays )
```

this subroutine writes the monthly reach output to the .sed file

##### Parameters

in	mdays	number of days simulated in month
----	-------	-----------------------------------

## 5.206 rsedyr.f90 File Reference

### Functions/Subroutines

- subroutine `rsedyr` (idlast)  
*this subroutine writes the yearly reach output to the .sed file*

#### 5.206.1 Detailed Description

file containing the subroutine rsedyr

##### Author

modified by Javier Burguete



## 5.206.2 Function/Subroutine Documentation

### 5.206.2.1 rsedyr()

```
subroutine rsedyr (
    integer, intent(in) idlast )
```

this subroutine writes the yearly reach output to the .sed file

#### Parameters

in	idlast	number of days simulated in month (none)
----	--------	--

## 5.207 rtbact.f90 File Reference

### Functions/Subroutines

- subroutine [rtbact](#) (jrch)  
*this subroutine routes bacteria through the stream network*

### 5.207.1 Detailed Description

file containing the subroutine rtbact

#### Author

modified by Javier Burguete

## 5.207.2 Function/Subroutine Documentation

### 5.207.2.1 rtbact()

```
subroutine rtbact (
    integer, intent(in) jrch )
```

this subroutine routes bacteria through the stream network

#### Parameters

in	jrch	reach number (none)
----	------	---------------------

## 5.208 rtday.f90 File Reference

### Functions/Subroutines

- subroutine [rtday](#) (jrch)

*this subroutine routes the daily flow through the reach using a variable storage coefficient*

### 5.208.1 Detailed Description

file containing the subroutine rtday

#### Author

modified by Javier Burguete

### 5.208.2 Function/Subroutine Documentation

#### 5.208.2.1 rtday()

```
subroutine rtday (
    integer, intent(in) jrch )
```

this subroutine routes the daily flow through the reach using a variable storage coefficient

#### Parameters

in	<i>jrch</i>	reach number
----	-------------	--------------

## 5.209 rteinit.f90 File Reference

### Functions/Subroutines

- subroutine [rteinit](#)

*This subroutine reads in the areas associated with files processed with the recday, recepic, recmon and recyear commands, calculates subbasin areas, calculates reach and hydrograph node drainage areas.*

### 5.209.1 Detailed Description

file containing the subroutine rteinit

#### Author

modified by Javier Burguete

## 5.210 rthmusk.f90 File Reference

### Functions/Subroutines

- subroutine [rthmusk](#) (i, jrch)

*this subroutine routes flow through a reach using the Muskingum method at a given time step*

### 5.210.1 Detailed Description

file containing the subroutine rthmusk

#### Author

code provided by Dr. Valentina Krysanova, Potsdam Institute for Climate Impact Research, Germany.  
Modified by N.Kannan, Blackland Research Center, Temple, USA.  
Modified by Javier Burguete

### 5.210.2 Function/Subroutine Documentation

#### 5.210.2.1 rthmusk()

```
subroutine rthmusk (
    integer, intent(in) i,
    integer, intent(in) jrch )
```

this subroutine routes flow through a reach using the Muskingum method at a given time step

#### Parameters

in	<i>i</i>	current day of simulation (none)
in	<i>jrch</i>	reach number

## 5.211 rthpest.f90 File Reference

### Functions/Subroutines

- subroutine [rthpest](#) (jrch)

*this subroutine computes the hourly stream pesticide balance (soluble and sorbed)*

### 5.211.1 Detailed Description

file containing the subroutine rthpest

**Author**

modified by Javier Burguete

**5.211.2 Function/Subroutine Documentation****5.211.2.1 rthpest()**

```
subroutine rthpest (
    integer, intent(in) jrch )
```

this subroutine computes the hourly stream pesticide balance (soluble and sorbed)

**Parameters**

in	jrch	reach number (none)
----	------	---------------------

**5.212 rthsed.f90 File Reference****Functions/Subroutines**

- subroutine [rthsed](#) (jrch)

*this subroutine routes sediment from subbasin to basin outlets on a sub-daily timestep Brownlie (1981) bed load model and Yang (1973, 1984) model added.*

**5.212.1 Detailed Description**

file containing the subroutine rthsed

**Author**

modified by J.Jeong and N.Kannan for urban sub-hourly sediment modeling,  
and by Balagi for bank erosion. Modified by Javier Burguete

**5.212.2 Function/Subroutine Documentation****5.212.2.1 rthsed()**

```
subroutine rthsed (
    integer, intent(in) jrch )
```

this subroutine routes sediment from subbasin to basin outlets on a sub-daily timestep Brownlie (1981) bed load model and Yang (1973, 1984) model added.

## Parameters

in	jrch	reach number (none)
----	------	---------------------

## 5.213 rthvsc.f90 File Reference

### Functions/Subroutines

- subroutine [rthvsc](#) (jrch)

*this subroutine routes flow at any required time step through the reach using a variable storage coefficient. Routing method: Enhanced Variable Storage routing (Jeong et al., 2014) adopted from APEX*

#### 5.213.1 Detailed Description

file containing the subroutine rthvsc

## Author

modified by Javier Burguete

#### 5.213.2 Function/Subroutine Documentation

##### 5.213.2.1 rthvsc()

```
subroutine rthvsc (
    integer, intent(in) jrch )
```

this subroutine routes flow at any required time step through the reach using a variable storage coefficient. Routing method: Enhanced Variable Storage routing (Jeong et al., 2014) adopted from APEX

## Parameters

in	jrch	reach number
----	------	--------------

## 5.214 rtmusk.f90 File Reference

### Functions/Subroutines

- subroutine [rtmusk](#) (i, jrch)

*this subroutine routes a daily flow through a reach using the Muskingum method*

### 5.214.1 Detailed Description

file containing the subroutine rtmusk

Author

modified by Javier Burguete

### 5.214.2 Function/Subroutine Documentation

#### 5.214.2.1 rtmusk()

```
subroutine rtmusk (
    integer, intent(in) i,
    integer, intent(in) jrch )
```

this subroutine routes a daily flow through a reach using the Muskingum method

Parameters

in	<i>i</i>	current day of simulation (none)
in	<i>jrch</i>	reach number

## 5.215 rtout.f90 File Reference

### Functions/Subroutines

- subroutine [rtout](#) (jrch)  
*this subroutine summarizes data for reaches*

#### 5.215.1 Detailed Description

file containing the subroutine rtout

Author

modified by Javier Burguete

### 5.215.2 Function/Subroutine Documentation

#### 5.215.2.1 rtout()

```
subroutine rtout (
    integer, intent(in) jrch )
```

this subroutine summarizes data for reaches

## Parameters

in	<i>jrch</i>	reach number (none)
----	-------------	---------------------

## 5.216 rtpest.f90 File Reference

### Functions/Subroutines

- subroutine [rtpest](#) (*jrch*)  
*this subroutine computes the daily stream pesticide balance (soluble and sorbed)*

#### 5.216.1 Detailed Description

file containing the subroutine `rtpest`

#### Author

modified by Javier Burguete

#### 5.216.2 Function/Subroutine Documentation

##### 5.216.2.1 `rtpest()`

```
subroutine rtpest (
    integer, intent(in) jrch )
```

this subroutine computes the daily stream pesticide balance (soluble and sorbed)

## Parameters

in	<i>jrch</i>	reach number (none)
----	-------------	---------------------

## 5.217 rtsed.f90 File Reference

### Functions/Subroutines

- subroutine [rtsed](#) (*jrch*)  
*this subroutine routes sediment from subbasin to basin outlets deposition is based on fall velocity and degradation on stream*

### 5.217.1 Detailed Description

file containing the subroutine rtsed

Author

modified by Javier Burguete

### 5.217.2 Function/Subroutine Documentation

#### 5.217.2.1 rtsed()

```
subroutine rtsed (
    integer, intent(in) jrch )
```

this subroutine routes sediment from subbasin to basin outlets deposition is based on fall velocity and degradation on stream

Parameters

in	jrch	reach number
----	------	--------------

## 5.218 rtsed2.f90 File Reference

### Functions/Subroutines

- subroutine [rtsed2](#) (jrch)

*this subroutine routes sediment from subbasin to basin outlets deposition is based on fall velocity and degradation on stream. Modification to the original SWAT sediment routine. Bagnolds strempower, Kodatie (Modified Simons-Li associates), Molinas&Wu strempower and Yang's sand-gravel equation approaches combined with Einstein's deposition equation plus particle size tracking*

#### 5.218.1 Detailed Description

file containing the subroutine rtsed2

Author

Balaji Narasimhan,  
Peter Allen,  
modified by Javier Burguete

#### 5.218.2 Function/Subroutine Documentation



**5.218.2.1 rtsed2()**

```
subroutine rtsed2 (
    integer, intent(in) jrch )
```

this subroutine routes sediment from subbasin to basin outlets deposition is based on fall velocity and degradation on stream. Modification to the original SWAT sediment routine. Bagnolds strempower, Kodatie (Modified Simons-Li associates), Molinas&Wu strempower and Yang's sand-gravel equation approaches combined with Einstein's deposition equation plus particle size tracking

**Parameters**

in	jrch	reach number
----	------	--------------

**5.219 sat\_excess.f90 File Reference****Functions/Subroutines**

- subroutine [sat\\_excess](#) (j1, j)  
*this subroutine is the master soil percolation component*

**5.219.1 Detailed Description**

file containing the subroutine sat\_excess

**Author**

modified by Javier Burguete

**5.219.2 Function/Subroutine Documentation****5.219.2.1 sat\_excess()**

```
subroutine sat_excess (
    integer, intent(in) j1,
    integer, intent(in) j )
```

this subroutine is the master soil percolation component

**Parameters**

in	j1	counter
in	j	HRU number

## 5.220 save.f90 File Reference

### Functions/Subroutines

- subroutine [save](#) (j)

*this subroutine writes daily records of loadings from a particular hydrograph storage location in the event output file. The save command is used when a watershed is broken into several individual runs and outflow from an upstream watershed needs to be stored for reading into a simulation of the downstream portion of the watershed. The recday command is used to read in the data.*

#### 5.220.1 Detailed Description

file containing the subroutine save

#### Author

modified by Javier Burguete

#### 5.220.2 Function/Subroutine Documentation

##### 5.220.2.1 save()

```
subroutine save (
    integer, intent(in) j )
```

this subroutine writes daily records of loadings from a particular hydrograph storage location in the event output file. The save command is used when a watershed is broken into several individual runs and outflow from an upstream watershed needs to be stored for reading into a simulation of the downstream portion of the watershed. The recday command is used to read in the data.

#### Parameters

in	j	file number (none)
----	---	--------------------

## 5.221 saveconc.f90 File Reference

### Functions/Subroutines

- subroutine [saveconc](#) (k)

*this subroutine saves hourly or average daily concentrations from a particular hydrograph node to a file*

#### 5.221.1 Detailed Description

file containing the subroutine saveconc

Author

modified by Javier Burguete

## 5.221.2 Function/Subroutine Documentation

### 5.221.2.1 saveconc()

```
subroutine saveconc (
    integer, intent(in) k )
```

this subroutine saves hourly or average daily concentrations from a particular hydrograph node to a file

Parameters

in	k	file number
----	---	-------------

## 5.222 sched\_mgt.f90 File Reference

### Functions/Subroutines

- subroutine [sched\\_mgt](#) (j)  
*this subroutine performs all management operations*

### 5.222.1 Detailed Description

file containing the subroutine sched\_mgt

Author

modified by Javier Burguete

## 5.222.2 Function/Subroutine Documentation

### 5.222.2.1 sched\_mgt()

```
subroutine sched_mgt (
    integer, intent(in) j )
```

this subroutine performs all management operations

## Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.223 schedule\_ops.f90 File Reference

### Functions/Subroutines

- subroutine [schedule\\_ops](#) (*j*)  
*this subroutine controls the simulation of the land phase of the hydrologic cycle*

#### 5.223.1 Detailed Description

file containing the subroutine schedule\_ops

#### Author

modified by Javier Burguete

#### 5.223.2 Function/Subroutine Documentation

##### 5.223.2.1 schedule\_ops()

```
subroutine schedule_ops (
    integer, intent(in) j )
```

this subroutine controls the simulation of the land phase of the hydrologic cycle

## Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.224 sim\_initday.f90 File Reference

### Functions/Subroutines

- subroutine [sim\\_initday](#)  
*this subroutine initialized arrays at the beginning of the day*

### 5.224.1 Detailed Description

file containing the subroutine sim\_initday

Author

modified by Javier Burguete

## 5.225 sim\_inityr.f90 File Reference

### Functions/Subroutines

- subroutine [sim\\_inityr](#)  
*this subroutine initializes variables at the beginning of the year*

### 5.225.1 Detailed Description

file containing the subroutine sim\_inityr

Author

modified by Javier Burguete

## 5.226 simulate.f90 File Reference

### Functions/Subroutines

- subroutine [simulate](#)  
*this subroutine contains the loops governing the modeling of processes in the watershed*

### 5.226.1 Detailed Description

file containing the subroutine simulate

Author

modified by Javier Burguete

## 5.227 slrgen.f90 File Reference

### Functions/Subroutines

- subroutine [slrgen](#) (j)  
*this subroutine generates solar radiation*

### 5.227.1 Detailed Description

file containing the subroutine slrgen

Author

modified by Javier Burguete

### 5.227.2 Function/Subroutine Documentation

#### 5.227.2.1 slrgen()

```
subroutine slrgen (
    integer, intent(in) j )
```

this subroutine generates solar radiation

Parameters

in	j	HRU number
----	---	------------

## 5.228 smeas.f90 File Reference

### Functions/Subroutines

- subroutine [smeas](#)

*this subroutine reads in daily solar radiation data and assigns the values to the proper HRUs*

#### 5.228.1 Detailed Description

file containing the subroutine smeas

Author

modified by Javier Burguete

## 5.229 snom.f90 File Reference

### Functions/Subroutines

- subroutine [snom](#) (j)

*this subroutine predicts daily snom melt when the average air temperature exceeds 0 degrees Celsius*

### 5.229.1 Detailed Description

file containing the subroutine snom

Author

modified by Javier Burguete

### 5.229.2 Function/Subroutine Documentation

#### 5.229.2.1 snom()

```
subroutine snom (
    integer, intent(in) j )
```

this subroutine predicts daily snom melt when the average air temperature exceeds 0 degrees Celsius

Parameters

in	j	HRU number
----	---	------------

## 5.230 soil\_chem.f90 File Reference

### Functions/Subroutines

- subroutine [soil\\_chem](#) (ii)  
*this subroutine initializes soil chemical properties*

#### 5.230.1 Detailed Description

file containing the subroutine soil\_chem

Author

modified by Javier Burguete

### 5.230.2 Function/Subroutine Documentation

#### 5.230.2.1 soil\_chem()

```
subroutine soil_chem (
    integer, intent(in) ii )
```

this subroutine initializes soil chemical properties

## Parameters

in	ii	HRU number
----	----	------------

## 5.231 soil\_phys.f90 File Reference

### Functions/Subroutines

- subroutine [soil\\_phys](#) (ii)  
*this subroutine initializes soil physical properties*

#### 5.231.1 Detailed Description

file containing the subroutine soil\_phys

## Author

modified by Javier Burguete

#### 5.231.2 Function/Subroutine Documentation

##### 5.231.2.1 soil\_phys()

```
subroutine soil_phys (  
    integer, intent(in) ii )
```

this subroutine initializes soil physical properties

## Parameters

in	ii	HRU number
----	----	------------

## 5.232 soil\_write.f90 File Reference

### Functions/Subroutines

- subroutine [soil\\_write](#) (i)  
*this subroutine writes output to the output.sol file*



### 5.232.1 Detailed Description

file containing the subroutine soil\_write

Author

modified by Javier Burguete

### 5.232.2 Function/Subroutine Documentation

#### 5.232.2.1 soil\_write()

```
subroutine soil_write (
    integer, intent(in) i )
```

this subroutine writes output to the output.sol file

Parameters

in	i	current day in simulation - loop counter (julian date)
----	---	--

## 5.233 solp.f90 File Reference

### Functions/Subroutines

- subroutine [solp](#) (j)

*this subroutine calculates the amount of phosphorus lost from the soil profile in runoff and the movement of soluble phosphorus from the first to the second layer via percolation*

#### 5.233.1 Detailed Description

file containing the subroutine solp

Author

modified by Javier Burguete

### 5.233.2 Function/Subroutine Documentation

#### 5.233.2.1 solp()

```
subroutine solp (
    integer, intent(in) j )
```

this subroutine calculates the amount of phosphorus lost from the soil profile in runoff and the movement of soluble phosphorus from the first to the second layer via percolation

## Parameters

<code>in</code>	<code>j</code>	HRU number (none)
-----------------	----------------	-------------------

## 5.234 solt.f90 File Reference

### Functions/Subroutines

- subroutine [solt](#) (`j`)

*this subroutine estimates daily average temperature at the bottom of each soil layer @parameter[in] j HRU number*

#### 5.234.1 Detailed Description

file containing the subroutine solt

## Author

modified by Javier Burguete

## 5.235 std1.f90 File Reference

### Functions/Subroutines

- subroutine [std1](#)

*this subroutine writes general information to the standard output file and header lines to miscellaneous output files*

#### 5.235.1 Detailed Description

file containing the subroutine std1

## Author

modified by Javier Burguete

## 5.236 std2.f90 File Reference

### Functions/Subroutines

- subroutine [std2](#)

*this subroutine writes general information to the standard output file and to miscellaneous output files*

### 5.236.1 Detailed Description

file containing the subroutine std2

Author

modified by Javier Burguete

## 5.237 std3.f90 File Reference

### Functions/Subroutines

- subroutine [std3](#)  
*this subroutine writes the annual table header to the standard output file*

### 5.237.1 Detailed Description

file containing the subroutine std3

Author

modified by Javier Burguete

## 5.238 stdaa.f90 File Reference

### Functions/Subroutines

- subroutine [stdaa](#)  
*this subroutine writes average annual output to .std file*

### 5.238.1 Detailed Description

file containing the subroutine stdaa

Author

modified by Javier Burguete

## 5.239 storeinitial.f90 File Reference

### Functions/Subroutines

- subroutine [storeinitial](#)  
*this subroutine saves initial values for variables that must be reset to rerun the simulation for different real time weather scenarios*

### 5.239.1 Detailed Description

file containing the subroutine storeinitial

Author

modified by Javier Burguete

## 5.240 structure.f90 File Reference

### Functions/Subroutines

- subroutine [structure](#) (k)  
*this subroutine adjusts dissolved oxygen content for aeration at structures.*

### 5.240.1 Detailed Description

file containing the subroutine structure

Author

A. Van Griensven, Hydrology-Vrije Universiteit Brussel, Belgium.  
Modified by Javier Burguete

### 5.240.2 Function/Subroutine Documentation

#### 5.240.2.1 structure()

```
subroutine structure (
    integer, intent(in) k )
```

this subroutine adjusts dissolved oxygen content for aeration at structures.

Parameters

in	k	reach number (none)
----	---	---------------------

## 5.241 sub\_subbasin.f90 File Reference

### Functions/Subroutines

- subroutine [sub\\_subbasin](#) (j)  
*this was split out from subbasin.f. Comments should be updated*

### 5.241.1 Detailed Description

file containing the subroutine sub\_subbasin

Author

modified by Javier Burguete

### 5.241.2 Function/Subroutine Documentation

#### 5.241.2.1 sub\_subbasin()

```
subroutine sub_subbasin (
    integer, intent(in) j )
```

this was split out from subbasin.f. Comments should be updated

Parameters

in	j	HRU number
----	---	------------

## 5.242 subaa.f90 File Reference

### Functions/Subroutines

- subroutine [subaa](#) (years)  
*this subroutine writes average annual subbasin output to the output.sub file*

#### 5.242.1 Detailed Description

file containing the subroutine subaa

Author

modified by Javier Burguete

### 5.242.2 Function/Subroutine Documentation

#### 5.242.2.1 subaa()

```
subroutine subaa (
    real*8, intent(in) years )
```

this subroutine writes average annual subbasin output to the output.sub file

## Parameters

in	years	length of simulation (years)
----	-------	------------------------------

## 5.243 subbasin.f90 File Reference

### Functions/Subroutines

- subroutine [subbasin](#) (i, sb)  
*this subroutine controls the simulation of the land phase of the hydrologic cycle*

#### 5.243.1 Detailed Description

file containing the subroutine subbasin

## Author

modified by Javier Burguete

#### 5.243.2 Function/Subroutine Documentation

##### 5.243.2.1 subbasin()

```
subroutine subbasin (
    integer, intent(in) i,
    integer, intent(inout) sb )
```

this subroutine controls the simulation of the land phase of the hydrologic cycle

## Parameters

in	i	current day in simulation—loop counter (julian date)
in, out	sb	subbasin number

## 5.244 subday.f90 File Reference

### Functions/Subroutines

- subroutine [subday](#) (j)  
*this subroutine writes daily subbasin output to the output.sub file*

### 5.244.1 Detailed Description

file containing the subroutine subday

Author

modified by Javier Burguete

### 5.244.2 Function/Subroutine Documentation

#### 5.244.2.1 subday()

```
subroutine subday (
    integer, intent(in) j )
```

this subroutine writes daily subbasin output to the output.sub file

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.245 submon.f90 File Reference

### Functions/Subroutines

- subroutine [submon](#)  
*this subroutine writes monthly subbasin output to the output.sub file*

#### 5.245.1 Detailed Description

file containing the subroutine submon

Author

modified by Javier Burguete

## 5.246 substor.f90 File Reference

### Functions/Subroutines

- subroutine [substor](#) (j)  
*this subroutine stores and lags lateral soil flow and nitrate*

### 5.246.1 Detailed Description

file containing the subroutine substor

Author

modified by Javier Burguete

### 5.246.2 Function/Subroutine Documentation

#### 5.246.2.1 substor()

```
subroutine substor (
    integer, intent(in) j )
```

this subroutine stores and lags lateral soil flow and nitrate

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.247 subwq.f90 File Reference

### Functions/Subroutines

- subroutine [subwq](#) (j)  
*this subroutine computes HRU loadings of chlorophyll-a, CBOD, and dissolved oxygen to the main channel*

#### 5.247.1 Detailed Description

file containing the subroutine subwq

Author

modified by Javier Burguete

#### 5.247.2 Function/Subroutine Documentation

##### 5.247.2.1 subwq()

```
subroutine subwq (
    integer, intent(in) j )
```

this subroutine computes HRU loadings of chlorophyll-a, CBOD, and dissolved oxygen to the main channel



## Parameters

<code>in</code>	<code>j</code>	HRU number (none)
-----------------	----------------	-------------------

## 5.248 subyr.f90 File Reference

### Functions/Subroutines

- subroutine `subyr`  
*this subroutine writes annual subbasin output to the output.sub file*

#### 5.248.1 Detailed Description

file containing the subroutine `subyr`

## Author

modified by Javier Burguete

## 5.249 sumhyd.f90 File Reference

### Functions/Subroutines

- subroutine `sumhyd`

#### 5.249.1 Detailed Description

file containing the subroutine `sumhyd`

## Author

modified by Javier Burguete

## 5.250 sumv.f90 File Reference

### Functions/Subroutines

- subroutine `sumv` (`j`)  
*this subroutine performs summary calculations for HRU*

### 5.250.1 Detailed Description

file containing the subroutine sumv

Author

modified by Javier Burguete

### 5.250.2 Function/Subroutine Documentation

#### 5.250.2.1 sumv()

```
subroutine sumv (
    integer, intent(in) j )
```

this subroutine performs summary calculations for HRU

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.251 surface.f90 File Reference

### Functions/Subroutines

- subroutine [surface](#) (i, j, sb)  
*this subroutine models surface hydrology at any desired time step*

#### 5.251.1 Detailed Description

file containing the subroutine surface

Author

modified by Javier Burguete

### 5.251.2 Function/Subroutine Documentation

#### 5.251.2.1 surface()

```
subroutine surface (
    integer, intent(in) i,
    integer, intent(in) j,
    integer, intent(in) sb )
```

this subroutine models surface hydrology at any desired time step

## Parameters

in	<i>i</i>	current day in simulation–loop counter (julian date)
in	<i>j</i>	HRU number (none)
in	<i>sb</i>	subbasin number (none)

## 5.252 surfst\_h2o.f90 File Reference

### Functions/Subroutines

- subroutine [surfst\\_h2o](#) (*j*)

*this subroutine determines the net surface runoff reaching the main channel on a given day. The net amount of water reaching the main channel can include water in surface runoff from the previous day and will exclude surface runoff generated on the current day which takes longer than one day to reach the main channel*

#### 5.252.1 Detailed Description

file containing the subroutine `surfst_h2o`

#### Author

modified by Javier Burguete

#### 5.252.2 Function/Subroutine Documentation

##### 5.252.2.1 `surfst_h2o()`

```
subroutine surfst_h2o (
    integer, intent(in) j )
```

this subroutine determines the net surface runoff reaching the main channel on a given day. The net amount of water reaching the main channel can include water in surface runoff from the previous day and will exclude surface runoff generated on the current day which takes longer than one day to reach the main channel

## Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.253 surfstor.f90 File Reference

### Functions/Subroutines

- subroutine [surfstor](#) (*j*, *sb*)

*this subroutine stores and lags sediment and nutrients in surface runoff*

### 5.253.1 Detailed Description

file containing the subroutine surfstor

Author

modified by Javier Burguete

### 5.253.2 Function/Subroutine Documentation

#### 5.253.2.1 surfstor()

```
subroutine surfstor (
    integer, intent(in) j,
    integer, intent(in) sb )
```

this subroutine stores and lags sediment and nutrients in surface runoff

Parameters

in	<i>j</i>	HRU number (none)
in	<i>sb</i>	subbasin number (none)

## 5.254 surq\_daycn.f90 File Reference

### Functions/Subroutines

- subroutine [surq\\_daycn](#) (*j*)  
*predicts daily runoff given daily precipitation and snow melt using a modified SCS curve number approach*

#### 5.254.1 Detailed Description

file containing the subroutine surq\_daycn

Author

modified by Javier Burguete

#### 5.254.2 Function/Subroutine Documentation

### 5.254.2.1 surq\_daycn()

```
subroutine surq_daycn (  
    integer, intent(in) j )
```

predicts daily runoff given daily precipitation and snow melt using a modified SCS curve number approach

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.255 surq\_greenampt.f90 File Reference

### Functions/Subroutines

- subroutine [surq\\_greenampt](#) (j)

*predicts daily runoff given breakpoint precipitation and snow melt using the Green & Ampt technique*

### 5.255.1 Detailed Description

file containing the subroutine surq\_greenampt

#### Author

modified by Javier Burguete

### 5.255.2 Function/Subroutine Documentation

#### 5.255.2.1 surq\_greenampt()

```
subroutine surq_greenampt (  
    integer, intent(in) j )
```

predicts daily runoff given breakpoint precipitation and snow melt using the Green & Ampt technique

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.256 swbl.f90 File Reference

### Functions/Subroutines

- subroutine [swbl](#) (snow, irrg)

*this subroutine checks the soil water balance at the end of the simulation*

### 5.256.1 Detailed Description

file containing the subroutine swbl

#### Author

modified by Javier Burguete

### 5.256.2 Function/Subroutine Documentation

#### 5.256.2.1 swbl()

```
subroutine swbl (
    real*8, intent(in) snow,
    real*8, intent(in) irrg )
```

this subroutine checks the soil water balance at the end of the simulation

#### Parameters

in	<i>snow</i>	snow in watershed at end of simulation
in	<i>irrg</i>	irrigation water applied to watershed

## 5.257 sweep.f90 File Reference

### Functions/Subroutines

- subroutine [sweep](#) (j)

*the subroutine performs the street sweeping operation*

### 5.257.1 Detailed Description

file containing the subroutine sweep

#### Author

modified by Javier Burguete

## 5.257.2 Function/Subroutine Documentation

### 5.257.2.1 sweep()

```
subroutine sweep (
    integer, intent(in) j )
```

the subroutine performs the street sweeping operation

#### Parameters

in	<i>j</i>	HRU number (none)
----	----------	-------------------

## 5.258 swu.f90 File Reference

### Functions/Subroutines

- subroutine [swu](#) (*j*)

*this subroutine distributes potential plant evaporation through the root zone and calculates actual plant water use based on soil water availability. Also estimates water stress factor*

### 5.258.1 Detailed Description

file containing the subroutine swu

#### Author

modified by Javier Burguete

## 5.258.2 Function/Subroutine Documentation

### 5.258.2.1 swu()

```
subroutine swu (
    integer, intent(in) j )
```

this subroutine distributes potential plant evaporation through the root zone and calculates actual plant water use based on soil water availability. Also estimates water stress factor

## Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.259 tair.f90 File Reference

### Functions/Subroutines

- real \*8 function [tair](#) (hr, jj)

*this function approximates hourly air temperature from daily max and min temperatures as documented by Campbell (1985)*

#### 5.259.1 Detailed Description

file containing the function tair

## Author

modified by Javier Burguete

#### 5.259.2 Function/Subroutine Documentation

##### 5.259.2.1 tair()

```
real*8 function tair (  
    integer, intent(in) hr,  
    integer, intent(in) jj )
```

this function approximates hourly air temperature from daily max and min temperatures as documented by Campbell (1985)

## Parameters

in	<i>hr</i>	hour of the day (none)
in	<i>jj</i>	HRU number (none)



**Returns**

air temperature for hour in HRU (deg C)

**5.260 tgen.f90 File Reference****Functions/Subroutines**

- subroutine [tgen](#) (j)

*this subroutine generates temperature data when the user chooses to simulate or when data is missing for particular days in the weather file*

**5.260.1 Detailed Description**

file containing the subroutine tgen

**Author**

modified by Javier Burguete

**5.260.2 Function/Subroutine Documentation****5.260.2.1 tgen()**

```
subroutine tgen (
    integer, intent(in) j )
```

this subroutine generates temperature data when the user chooses to simulate or when data is missing for particular days in the weather file

**Parameters**

in	j	HRU number
----	---	------------

**5.261 theta.f90 File Reference****Functions/Subroutines**

- real \*8 function [theta](#) (r20, thk, tmp)

*this function corrects rate constants for temperature. Equation is III-52 from QUAL2E*

### 5.261.1 Detailed Description

file containing the function theta

Author

modified by Javier Burguete

### 5.261.2 Function/Subroutine Documentation

#### 5.261.2.1 theta()

```
real*8 function theta (
    real*8, intent(in) r20,
    real*8, intent(in) thk,
    real*8, intent(in) tmp )
```

this function corrects rate constants for temperature. Equation is III-52 from QUAL2E

Parameters

in	<i>r20</i>	value of the reaction rate coefficient at the standard temperature (20 degrees C) (1/day)
in	<i>thk</i>	temperature adjustment factor (empirical constant for each reaction coefficient) (none)
in	<i>tmp</i>	temperature on current day (deg C)

Returns

value of the reaction rate coefficient at the local temperature (1/day)

## 5.262 tillfactor.f90 File Reference

### Functions/Subroutines

- subroutine [tillfactor](#) (j, bmix, emix, dtil, sol\_thick)

*this procedure increases tillage factor (tillagef(l,j) per layer for each operation. The tillage factor settling will depend of soil moisture (tentatively) and must be called every day. For simplicity the settling is calculated now at the soil carbon sub because soil water content is available.*

#### 5.262.1 Detailed Description

file containing the subroutine tillfactor

Author

modified by Javier Burguete

## 5.262.2 Function/Subroutine Documentation

### 5.262.2.1 tillfactor()

```
subroutine tillfactor (
    integer, intent(in) j,
    real*8, intent(in) bmix,
    real*8, intent(inout) emix,
    real*8, intent(in) dtil,
    real*8, dimension(sol_nly(j)), intent(in) sol_thick )
```

this procedure increases tillage factor (tillagef(l,j) per layer for each operation. The tillage factor settling will depend of soil moisture (tentatively) and must be called every day. For simplicity the settling is calculated now at the soil carbon sub because soil water content is available.

#### Parameters

in	<i>j</i>	HRU number (none)
in	<i>bmix</i>	biological mixing efficiency: this number is zero for tillage operations (none)
in, out	<i>emix</i>	mixing efficiency (none)
in	<i>dtil</i>	depth of mixing (mm)
in	<i>sol_thick</i>	<p>The tillage factor depends on the cumulative soil disturbance rating = csdr For simplicity, csdr is a function of emix. First step is to calculate "current" csdr by inverting tillage factor function. The effect of texture on tillage factor (ZZ) is removed first (and recovered at the end of the procedure).</p> $YY = \text{tillagef}(l, j) / ZZ$ <p>Since the tillage factor function is non linear, iterations are needed. <math>XX = 0.5</math> is the initial value that works OK for the range of values observed. If a layer is only partially tilled then emix is corrected accordingly</p>

## 5.263 tmeas.f90 File Reference

### Functions/Subroutines

- subroutine [tmeas](#)

*this subroutine reads in temperature data and assigns it to the HRUs*

### 5.263.1 Detailed Description

file containing the subroutine tmeas

#### Author

modified by Javier Burguete

## 5.264 tran.f90 File Reference

### Functions/Subroutines

- subroutine `tran` (j)  
*this subroutine computes tributary channel transmission losses*

#### 5.264.1 Detailed Description

file containing the subroutine tran

##### Author

modified by Javier Burguete

#### 5.264.2 Function/Subroutine Documentation

##### 5.264.2.1 tran()

```
subroutine tran (
    integer, intent(in) j )
```

this subroutine computes tributary channel transmission losses

##### Parameters

in	<i>j</i>	HRU number (none)
----	----------	-------------------

## 5.265 transfer.f90 File Reference

### Functions/Subroutines

- subroutine `transfer` (j)  
*this subroutine transfers water*

#### 5.265.1 Detailed Description

file containing the subroutine transfer

##### Author

modified by Javier Burguete

## 5.265.2 Function/Subroutine Documentation

### 5.265.2.1 transfer()

```
subroutine transfer (
    integer, intent(in) j )
```

this subroutine transfers water

#### Parameters

in	<i>j</i>	reach or reservoir # from which water is removed (none)
----	----------	---

## 5.266 tstr.f90 File Reference

### Functions/Subroutines

- subroutine [tstr](#) (*j*)  
*computes temperature stress for crop growth - strstmp*

### 5.266.1 Detailed Description

file containing the subroutine tstr

#### Author

modified by Javier Burguete

## 5.266.2 Function/Subroutine Documentation

### 5.266.2.1 tstr()

```
subroutine tstr (
    integer, intent(in) j )
```

computes temperature stress for crop growth - strstmp

#### Parameters

in	<i>j</i>	HRU number
----	----------	------------

## 5.267 ttcoef.f90 File Reference

### Functions/Subroutines

- subroutine [ttcoef](#) (k)

*this subroutine computes travel time coefficients for routing along the main channel*

#### 5.267.1 Detailed Description

file containing the subroutine ttcoef

##### Author

modified by Javier Burguete

#### 5.267.2 Function/Subroutine Documentation

##### 5.267.2.1 ttcoef()

```
subroutine ttcoef (
    integer, intent(in) k )
```

this subroutine computes travel time coefficients for routing along the main channel

##### Parameters

in	k	HRU number
----	---	------------

## 5.268 ttcoef\_wway.f90 File Reference

### Functions/Subroutines

- subroutine [ttcoef\\_wway](#) (j)

*this subroutine computes travel time coefficients for routing along the main channel - grassed waterways*

#### 5.268.1 Detailed Description

file containing the subroutine ttcoef\_wway

##### Author

modified by Javier Burguete

## 5.269 urb\_bmp.f90 File Reference

### Functions/Subroutines

- subroutine [urb\\_bmp](#) (j)  
*this subroutine*

#### 5.269.1 Detailed Description

file containing the subroutine urb\_bmp

#### Author

modified by Javier Burguete

#### 5.269.2 Function/Subroutine Documentation

##### 5.269.2.1 urb\_bmp()

```
subroutine urb_bmp (
    integer, intent(in) j )
```

this subroutine

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.270 urban.f90 File Reference

### Functions/Subroutines

- subroutine [urban](#) (j)  
*this subroutine computes loadings from urban areas using the USGS regression equations or a build-up/wash-off algorithm*

#### 5.270.1 Detailed Description

file containing the subroutine urban

#### Author

modified by Javier Burguete

## 5.270.2 Function/Subroutine Documentation

### 5.270.2.1 urban()

```
subroutine urban (
    integer, intent(in) j )
```

this subroutine computes loadings from urban areas using the USGS regression equations or a build-up/wash-off algorithm

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.271 urbanhr.f90 File Reference

### Functions/Subroutines

- subroutine [urbanhr](#) (j)

*this subroutine computes loadings from urban areas using the a build-up/wash-off algorithm at subdaily time intervals*

### 5.271.1 Detailed Description

file containing the subroutine urbanhr

#### Author

modified by Javier Burguete

## 5.271.2 Function/Subroutine Documentation

### 5.271.2.1 urbanhr()

```
subroutine urbanhr (
    integer, intent(in) j )
```

this subroutine computes loadings from urban areas using the a build-up/wash-off algorithm at subdaily time intervals



## Parameters

<code>in</code>	<code>j</code>	HRU number (none)
-----------------	----------------	-------------------

## 5.272 varinit.f90 File Reference

### Functions/Subroutines

- subroutine [varinit](#) (`j`)  
*this subroutine initializes variables for the daily simulation of the land phase of the hydrologic cycle (the subbasin command loop)*

#### 5.272.1 Detailed Description

file containing the subroutine varinit

#### Author

modified by Javier Burguete

#### 5.272.2 Function/Subroutine Documentation

##### 5.272.2.1 varinit()

```
subroutine varinit (
    integer, intent(in) j )
```

this subroutine initializes variables for the daily simulation of the land phase of the hydrologic cycle (the subbasin command loop)

## Parameters

<code>in</code>	<code>j</code>	HRU number
-----------------	----------------	------------

## 5.273 vbl.f90 File Reference

### Functions/Subroutines

- subroutine [vbl](#) (`evx`, `spx`, `pp`, `qin`, `ox`, `vx1`, `vy`, `yi`, `yo`, `ysx`, `vf`, `vyf`, `aha`)  
*this subroutine checks the water and sediment balance for ponds and reservoirs at the end of a simulation*

### 5.273.1 Detailed Description

file containing the subroutine vbl

Author

modified by Javier Burguete

### 5.273.2 Function/Subroutine Documentation

#### 5.273.2.1 vbl()

```
subroutine vbl (
    real*8, intent(in) evx,
    real*8, intent(in) spx,
    real*8, intent(in) pp,
    real*8, intent(in) qin,
    real*8, intent(in) ox,
    real*8, intent(inout) vx1,
    real*8, intent(inout) vy,
    real*8, intent(in) yi,
    real*8, intent(in) yo,
    real*8, intent(in) ysx,
    real*8, intent(in) vf,
    real*8, intent(in) vyf,
    real*8, intent(in) aha )
```

this subroutine checks the water and sediment balance for ponds and reservoirs at the end of a simulation

#### Parameters

in	<i>evx</i>	evaporation from water body
in	<i>spx</i>	seepage from water body
in	<i>pp</i>	precipitation on water body
in	<i>qin</i>	water entering water body
in	<i>ox</i>	water leaving water body
in, out	<i>vx1</i>	(in) volume of water in water body at beginning of simulation (out) dfw expressed as depth over drainage area
in, out	<i>vy</i>	(in) sediment in water body at beginning of simulation (out) dfy expressed as loading per unit area for drainage area
in	<i>yi</i>	sediment entering water body
in	<i>yo</i>	sediment leaving water body
in	<i>ysx</i>	change in sediment level in water body
in	<i>vf</i>	volume of water in water body at end of simulation
in	<i>vyf</i>	sediment in water body at end of simulation
in	<i>aha</i>	area draining into water body

## 5.274 virtual.f90 File Reference

### Functions/Subroutines

- subroutine [virtual](#) (i, j, k, sb)

*this subroutine summarizes data for subbasins with multiple HRUs and prints the daily output.hru file*

### 5.274.1 Detailed Description

file containing the subroutine virtual

#### Author

modified by Javier Burguete

### 5.274.2 Function/Subroutine Documentation

#### 5.274.2.1 virtual()

```
subroutine virtual (
    integer, intent(in) i,
    integer, intent(in) j,
    integer, intent(in) k,
    integer, intent(in) sb )
```

this subroutine summarizes data for subbasins with multiple HRUs and prints the daily output.hru file

#### Parameters

in	<i>i</i>	current day in simulation—loop counter (julian date)
in	<i>j</i>	HRU number
in	<i>k</i>	
in	<i>sb</i>	subbasin number

## 5.275 volq.f90 File Reference

### Functions/Subroutines

- subroutine [volq](#) (j)

*call subroutines to calculate the current day's CN for the HRU and to calculate surface runoff*

### 5.275.1 Detailed Description

file containing the subroutine volq

Author

modified by Javier Burguete

### 5.275.2 Function/Subroutine Documentation

#### 5.275.2.1 volq()

```
subroutine volq (
    integer, intent(in) j )
```

call subroutines to calculate the current day's CN for the HRU and to calculate surface runoff

Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.276 washp.f90 File Reference

### Functions/Subroutines

- subroutine [washp](#) (j)  
*this subroutine calculates the amount of pesticide washed off the plant foliage and onto the soil*

#### 5.276.1 Detailed Description

file containing the subroutine washp

Author

modified by Javier Burguete Tolosa

### 5.276.2 Function/Subroutine Documentation

#### 5.276.2.1 washp()

```
subroutine washp (
    integer, intent(in) j )
```

this subroutine calculates the amount of pesticide washed off the plant foliage and onto the soil

## Parameters

in	j	HRU number
----	---	------------

## 5.277 watbal.f90 File Reference

### Functions/Subroutines

- subroutine [watbal](#) (j)

*this subroutine computes the daily water balance for each HRU changes in storage should equal water losses from the system write statements can be uncommented for model debugging. This subroutine will give errors for HRUs receiving irrigation water from reaches or reservoirs*

#### 5.277.1 Detailed Description

file containing the subroutine watbal

#### Author

modified by Javier Burguete

#### 5.277.2 Function/Subroutine Documentation

##### 5.277.2.1 watbal()

```
subroutine watbal (
    integer, intent(in) j )
```

this subroutine computes the daily water balance for each HRU changes in storage should equal water losses from the system write statements can be uncommented for model debugging. This subroutine will give errors for HRUs receiving irrigation water from reaches or reservoirs

## Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.278 water\_hru.f90 File Reference

### Functions/Subroutines

- subroutine [water\\_hru](#) (j)

*this subroutine compute pet and et using Priestly-Taylor and a coefficient*

### 5.278.1 Detailed Description

file containing the subroutine `water_hru`

Author

modified by Javier Burguete

### 5.278.2 Function/Subroutine Documentation

#### 5.278.2.1 `water_hru()`

```
subroutine water_hru (
    integer, intent(in) j )
```

this subroutine compute pet and et using Priestly-Taylor and a coefficient

Parameters

<code>in</code>	<code>j</code>	HRU number
-----------------	----------------	------------

## 5.279 `watqual.f90` File Reference

### Functions/Subroutines

- subroutine [watqual](#) (`i`, `jrch`, `k`)  
*this subroutine performs in-stream nutrient transformations and water quality calculations*

#### 5.279.1 Detailed Description

file containing the subroutine `watqual`

Author

modified by Javier Burguete

### 5.279.2 Function/Subroutine Documentation

#### 5.279.2.1 `watqual()`

```
subroutine watqual (
    integer, intent(in) i,
    integer, intent(in) jrch,
    integer, intent(in) k )
```

this subroutine performs in-stream nutrient transformations and water quality calculations

## Parameters

in	<i>i</i>	current day in simulation–loop counter (julian date)
in	<i>jrch</i>	reach number (none)
in	<i>k</i>	inflow hydrograph storage location number (none)

## 5.280 watqual2.f90 File Reference

### Functions/Subroutines

- subroutine [watqual2](#) (jrch, k)  
*this subroutine performs in-stream nutrient transformations and water quality calculations*

#### 5.280.1 Detailed Description

file containing the subroutine watqual2

#### Author

adapted by Ann van Griensven, Belgium.  
Modified by Javier Burguete

#### 5.280.2 Function/Subroutine Documentation

##### 5.280.2.1 watqual2()

```
subroutine watqual2 (
    integer, intent(in) jrch,
    integer, intent(in) k )
```

this subroutine performs in-stream nutrient transformations and water quality calculations

## Parameters

in	<i>jrch</i>	reach number (none)
in	<i>k</i>	inflow hydrograph storage location number (none)

## 5.281 watable.f90 File Reference

### Functions/Subroutines

- subroutine [watable](#) (j)

*this subroutine is the master soil percolation component. param[in] j HRU number*

### 5.281.1 Detailed Description

file containing the subroutine wattable

Author

modified by Javier Burguete

## 5.282 watuse.f90 File Reference

### Functions/Subroutines

- subroutine [watuse](#) (j)

*this subroutine removes water from appropriate source (pond, shallow aquifer, and/or deep aquifer) for consumptive water use*

### 5.282.1 Detailed Description

file containing the subroutine watuse

Author

modified by Javier Burguete

### 5.282.2 Function/Subroutine Documentation

#### 5.282.2.1 watuse()

```
subroutine watuse (
    integer, intent(in) j )
```

this subroutine removes water from appropriate source (pond, shallow aquifer, and/or deep aquifer) for consumptive water use

Parameters

in	j	HRU number (none)
----	---	-------------------



## 5.283 weatgn.f90 File Reference

### Functions/Subroutines

- subroutine [weatgn](#) (j)

*this subroutine generates weather parameters used to simulate the impact of precipitation on the other climatic processes*

#### 5.283.1 Detailed Description

file containing the subroutine weatgn

##### Author

modified by Javier Burguete

#### 5.283.2 Function/Subroutine Documentation

##### 5.283.2.1 weatgn()

```
subroutine weatgn (  
    integer, intent(in) j )
```

this subroutine generates weather parameters used to simulate the impact of precipitation on the other climatic processes

##### Parameters

in	j	HRU number
----	---	------------

## 5.284 wetlan.f90 File Reference

### Functions/Subroutines

- subroutine [wetlan](#) (j)

*this subroutine simulates wetlands*

#### 5.284.1 Detailed Description

file containing the subroutine wetlan

##### Author

modified by Javier Burguete

## 5.284.2 Function/Subroutine Documentation

### 5.284.2.1 wetlan()

```
subroutine wetlan (
    integer, intent(in) j )
```

this subroutine simulates wetlands

#### Parameters

in	j	HRU number (none)
----	---	-------------------

## 5.285 wmeas.f90 File Reference

### Functions/Subroutines

- subroutine [wmeas](#)  
*this subroutine reads in wind speed data from file and assigns the data to HRUs*

### 5.285.1 Detailed Description

file containing the subroutine wmeas

#### Author

modified by Javier Burguete

## 5.286 wndgen.f90 File Reference

### Functions/Subroutines

- subroutine [wndgen](#) (j)  
*this subroutine generates wind speed*

### 5.286.1 Detailed Description

file containing the subroutine wndgen

#### Author

modified by Javier Burguete

## 5.286.2 Function/Subroutine Documentation

### 5.286.2.1 wndgen()

```
subroutine wndgen (
    integer, intent(in) j )
```

this subroutine generates wind speed

#### Parameters

in	j	HRU number
----	---	------------

## 5.287 writea.f90 File Reference

### Functions/Subroutines

- subroutine [writea](#) (i)  
*this subroutine writes annual output*

### 5.287.1 Detailed Description

file containing the subroutine writea

#### Author

modified by Javier Burguete

## 5.287.2 Function/Subroutine Documentation

### 5.287.2.1 writea()

```
subroutine writea (
    integer, intent(in) i )
```

this subroutine writes annual output

#### Parameters

in	i	current day of simulation (julian date)
----	---	---

## 5.288 writeaa.f90 File Reference

### Functions/Subroutines

- subroutine [writeaa](#)  
*this subroutine writes average annual output*

#### 5.288.1 Detailed Description

file containing the subroutine writeaa

##### Author

modified by Javier Burguete

## 5.289 writed.f90 File Reference

### Functions/Subroutines

- subroutine [writed](#)  
*this subroutine contains the daily output writes*

#### 5.289.1 Detailed Description

file containing the subroutine writed

##### Author

modified by Javier Burguete

## 5.290 writem.f90 File Reference

### Functions/Subroutines

- subroutine [writem](#) (i)  
*this subroutine writes monthly output*

#### 5.290.1 Detailed Description

file containing the subroutine writem

##### Author

modified by Javier Burguete

#### 5.290.2 Function/Subroutine Documentation

##### 5.290.2.1 writem()

```
subroutine writem (  
    integer, intent(in) i )
```

this subroutine writes monthly output

## Parameters

in	i	current day of simulation (julian date)
----	---	---

## 5.291 xmon.f90 File Reference

### Functions/Subroutines

- subroutine [xmon](#)  
*this subroutine determines the month, given the julian date and leap year flag*

#### 5.291.1 Detailed Description

file containing the subroutine xmon

#### Author

modified by Javier Burguete

## 5.292 ysed.f90 File Reference

### Functions/Subroutines

- subroutine [ysed](#) (iwave, j)  
*this subroutine predicts daily soil loss caused by water erosion using the modified universal soil loss equation*

#### 5.292.1 Detailed Description

file containing the subroutine ysed

#### Author

modified by Javier Burguete

#### 5.292.2 Function/Subroutine Documentation

##### 5.292.2.1 ysed()

```
subroutine ysed (
    integer, intent(in) iwave,
    integer, intent(in) j )
```

this subroutine predicts daily soil loss caused by water erosion using the modified universal soil loss equation

## Parameters

in	<i>iwave</i>	flag to differentiate calculation of HRU and subbasin sediment calculation (none) iwave = 0 for HRU iwave = subbasin # for subbasin
in	<i>j</i>	HRU number

## 5.293 zero0.f90 File Reference

### Functions/Subroutines

- subroutine [zero0](#)  
*this subroutine initializes the values for some of the arrays*

#### 5.293.1 Detailed Description

file containing the subroutine zero0

## Author

modified by Javier Burguete

## 5.294 zero1.f90 File Reference

### Functions/Subroutines

- subroutine [zero1](#)  
*this subroutine initializes the values for some of the arrays*

#### 5.294.1 Detailed Description

file containing the subroutine zero1

## Author

modified by Javier Burguete

## 5.295 zero2.f90 File Reference

### Functions/Subroutines

- subroutine [zero2](#)  
*this subroutine zeros all array values*

### 5.295.1 Detailed Description

file containing the subroutine zero2

Author

modified by Javier Burguete

## 5.296 zero\_urbn.f90 File Reference

### Functions/Subroutines

- subroutine [zero\\_urbn](#)  
*this subroutine zeros all array values used in urban modeling*

### 5.296.1 Detailed Description

file containing the subroutine zero\_urbn

Author

modified by Javier Burguete

## 5.297 zeroini.f90 File Reference

### Functions/Subroutines

- subroutine [zeroini](#)  
*this subroutine zeros values for single array variables*

### 5.297.1 Detailed Description

file containing the subroutine zeroini

Author

modified by Javier Burguete





# Bibliography

- [1] P Bratley, B L Fox, and L E Schrage. *A Guide to Simulation*. Springer-Verlag, New York, USA, 1983. [108](#)
- [2] Armen R Kemanian and Claudio O Stöckle. C-farm: A simple model to evaluate the carbon balance of soil profiles. *European Journal of Agronomy*, 32(1):22–29, 2010. [116](#), [117](#)
- [3] J. E. McCray, S. L. Kirkland, R. L. Siegrist, and G. D. Thyne. Model parameters for simulating fate and transport of on-site wastewater nutrients. *Ground Water*, 43(4):628–639, 2005. [297](#)
- [4] R. L. Siegrist, J. McCray, L. Weintraub, C. Chen, J. Bagdol, P. Lemonds, S. Van Cuyk, K. Lowe, R. Goldstein, and J. Rada. Quantifying site-scale processes and watershed-scale cumulative effects of decentralized wastewater systems, project no. wu-ht-00-27. Prepared for the National Decentralized Water Resources Capacity Development Project, Washington University, St. Louis, MO, by the Colorado School of Mines, 2005. [110](#), [297](#)
- [5] P.A. Vadas and Michael White. Validating soil phosphorus routines in the swat model. *Transactions of ASABE*, 53, 09 2010. [274](#)



# Index

addh  
    addh.f90, 101  
addh.f90, 101  
    addh, 101  
albedo  
    albedo.f90, 102  
albedo.f90, 102  
    albedo, 102  
allocate\_parms.f90, 102  
alph  
    alph.f90, 103  
alph.f90, 103  
    alph, 103  
anfert  
    anfert.f90, 104  
anfert.f90, 103  
    anfert, 104  
apex\_day  
    apex\_day.f90, 104  
apex\_day.f90, 104  
    apex\_day, 104  
apply  
    apply.f90, 105  
apply.f90, 105  
    apply, 105  
ascrv  
    ascrv.f90, 106  
ascrv.f90, 106  
    ascrv, 106  
atri  
    atri.f90, 107  
atri.f90, 107  
    atri, 107  
aunif  
    aunif.f90, 108  
aunif.f90, 107  
    aunif, 108  
autoirr  
    autoirr.f90, 109  
autoirr.f90, 108  
    autoirr, 109  
  
bacteria  
    bacteria.f90, 109  
bacteria.f90, 109  
    bacteria, 109  
biozone  
    biozone.f90, 110  
biozone.f90, 110  
    biozone, 110  
  
bmp\_det\_pond  
    bmp\_det\_pond.f90, 111  
bmp\_det\_pond.f90, 110  
    bmp\_det\_pond, 111  
bmp\_ri\_pond.f90, 111  
bmp\_sand\_filter.f90, 111  
bmp\_sed\_pond.f90, 112  
bmp\_wet\_pond  
    bmp\_wet\_pond.f90, 112  
bmp\_wet\_pond.f90, 112  
    bmp\_wet\_pond, 112  
    pipe\_discharge, 113  
bmpinit  
    bmpinit.f90, 113  
bmpinit.f90, 113  
    bmpinit, 113  
buffer  
    buffer.f90, 114  
buffer.f90, 114  
    buffer, 114  
burnop  
    burnop.f90, 115  
burnop.f90, 114  
    burnop, 115  
  
canopyint  
    canopyint.f90, 115  
canopyint.f90, 115  
    canopyint, 115  
caps  
    caps.f90, 116  
caps.f90, 116  
    caps, 116  
carbon  
    carbon\_new.f90, 117  
carbon\_new.f90, 116  
    carbon, 117  
carbon\_zhang2  
    carbon\_zhang2.f90, 118  
carbon\_zhang2.f90, 117  
    carbon\_zhang2, 118  
cfactor  
    cfactor.f90, 118  
cfactor.f90, 118  
    cfactor, 118  
clgen  
    clgen.f90, 119  
clgen.f90, 119  
    clgen, 119  
clicon

- clicon.f90, 120
- clicon.f90, 119
  - clicon, 120
- command
  - command.f90, 120
- command.f90, 120
  - command, 120
- conapply
  - conapply.f90, 121
- conapply.f90, 121
  - conapply, 121
- confert
  - confert.f90, 122
- confert.f90, 122
  - confert, 122
- crackflow
  - crackflow.f90, 123
- crackflow.f90, 122
  - crackflow, 123
- crackvol
  - crackvol.f90, 123
- crackvol.f90, 123
  - crackvol, 123
- curno
  - curno.f90, 124
- curno.f90, 124
  - curno, 124
- dailycn
  - dailycn.f90, 125
- dailycn.f90, 124
  - dailycn, 125
- decay
  - decay.f90, 125
- decay.f90, 125
  - decay, 125
- depstor
  - depstor.f90, 126
- depstor.f90, 126
  - depstor, 126
- distributed\_bmps.f90, 126
- dormant
  - dormant.f90, 127
- dormant.f90, 127
  - dormant, 127
- drains
  - drains.f90, 128
- drains.f90, 127
  - drains, 128
- dstn1
  - dstn1.f90, 128
- dstn1.f90, 128
  - dstn1, 128
- ee
  - ee.f90, 130
- ee.f90, 130
  - ee, 130
- eiusle.f90, 131
- enrsb
  - enrsb.f90, 131
- enrsb.f90, 131
  - enrsb, 131
- estimate\_ksat
  - estimate\_ksat.f90, 132
- estimate\_ksat.f90, 132
  - estimate\_ksat, 132
- etact.f90, 133
- etpot
  - etpot.f90, 133
- etpot.f90, 133
  - etpot, 133
- expo
  - expo.f90, 134
- expo.f90, 134
  - expo, 134
- fcgd.f90, 134
- fert
  - fert.f90, 135
- fert.f90, 135
  - fert, 135
- filter
  - filter.f90, 136
- filter.f90, 135
  - filter, 136
- filtw
  - filtw.f90, 136
- filtw.f90, 136
  - filtw, 136
- finalbal.f90, 137
- gcycl.f90, 137
- getallo.f90, 137
- grass\_wway
  - grass\_wway.f90, 138
- grass\_wway.f90, 138
  - grass\_wway, 138
- graze
  - graze.f90, 139
- graze.f90, 138
  - graze, 139
- grow
  - grow.f90, 139
- grow.f90, 139
  - grow, 139
- gw\_no3
  - gw\_no3.f90, 140
- gw\_no3.f90, 140
  - gw\_no3, 140
- gwmod
  - gwmod.f90, 141
- gwmod.f90, 140
  - gwmod, 141
- gwmod\_deep
  - gwmod\_deep.f90, 141
- gwmod\_deep.f90, 141
  - gwmod\_deep, 141

- gwnutr
  - gwnutr.f90, 142
- gwnutr.f90, 142
  - gwnutr, 142
- h2omgt\_init.f90, 142
- harvestop
  - harvestop.f90, 143
- harvestop.f90, 143
  - harvestop, 143
- harvkillop
  - harvkillop.f90, 144
- harvkillop.f90, 143
  - harvkillop, 144
- headout.f90, 144
- hhnoqual
  - hhnoqual.f90, 145
- hhnoqual.f90, 144
  - hhnoqual, 145
- hhwatqual
  - hhwatqual.f90, 145
- hhwatqual.f90, 145
  - hhwatqual, 145
- hmeas.f90, 146
- HQDAV.f90, 146
- hruaa
  - hruaa.f90, 147
- hruaa.f90, 146
  - hruaa, 147
- hruallo.f90, 147
- hruday
  - hruday.f90, 148
- hruday.f90, 147
  - hruday, 148
- hrumon.f90, 148
- hrupond
  - hrupond.f90, 149
- hrupond.f90, 148
  - hrupond, 149
- hrupondhr
  - hrupondhr.f90, 149
- hrupondhr.f90, 149
  - hrupondhr, 149
- hruyr.f90, 150
- hydroinit.f90, 150
- icl
  - icl.f90, 151
- icl.f90, 150
  - icl, 151
- igropt
  - parm, 99
- impnd\_init.f90, 151
- impndaa
  - impndaa.f90, 152
- impndaa.f90, 151
  - impndaa, 152
- impndday
  - impndday.f90, 152
- impndday.f90, 152
  - impndday, 152
- impndmon.f90, 153
- impndyr.f90, 153
- irr\_rch
  - irr\_rch.f90, 154
- irr\_rch.f90, 153
  - irr\_rch, 154
- irr\_res
  - irr\_res.f90, 154
- irr\_res.f90, 154
  - irr\_res, 154
- irrigate
  - irrigate.f90, 155
- irrigate.f90, 155
  - irrigate, 155
- irrsb
  - irrsb.f90, 156
- irrsb.f90, 155
  - irrsb, 156
- jdt
  - jdt.f90, 156
- jdt.f90, 156
  - jdt, 156
- killop
  - killop.f90, 158
- killop.f90, 158
  - killop, 158
- lakeq
  - lakeq.f90, 159
- lakeq.f90, 158
  - lakeq, 159
- latsed
  - latsed.f90, 159
- latsed.f90, 159
  - latsed, 159
- layersplit.f90, 160
- lid\_cistern
  - lid\_cistern.f90, 160
- lid\_cistern.f90, 160
  - lid\_cistern, 160
- lid\_greenroof
  - lid\_greenroof.f90, 161
- lid\_greenroof.f90, 161
  - lid\_greenroof, 161
- lid\_por pavement
  - lid\_por pavement.f90, 162
- lid\_por pavement.f90, 162
  - lid\_por pavement, 162
- lid\_rain garden
  - lid\_rain garden.f90, 163
- lid\_rain garden.f90, 162
  - lid\_rain garden, 163
- lidinit
  - lidinit.f90, 164
- lidinit.f90, 163

- lidinit, 164
- lids
  - lids.f90, 164
- lids.f90, 164
  - lids, 164
- log\_normal
  - log\_normal.f90, 165
- log\_normal.f90, 165
  - log\_normal, 165
- lwqdef
  - lwqdef.f90, 166
- lwqdef.f90, 166
  - lwqdef, 166
- main.f90, 166
- modparm.f90, 167
- ndenit
  - ndenit.f90, 251
- ndenit.f90, 250
  - ndenit, 251
- newtillmix
  - newtillmix.f90, 251
- newtillmix.f90, 251
  - newtillmix, 251
- nfix
  - nfix.f90, 252
- nfix.f90, 252
  - nfix, 252
- nitvol
  - nitvol.f90, 253
- nitvol.f90, 253
  - nitvol, 253
- nlch
  - nlch.f90, 254
- nlch.f90, 253
  - nlch, 254
- nminrl
  - nminrl.f90, 254
- nminrl.f90, 254
  - nminrl, 254
- noqual
  - noqual.f90, 255
- noqual.f90, 255
  - noqual, 255
- npup
  - npup.f90, 256
- npup.f90, 255
  - npup, 256
- nrain
  - nrain.f90, 256
- nrain.f90, 256
  - nrain, 256
- nup
  - nup.f90, 257
- nup.f90, 257
  - nup, 257
- nuts
  - nuts.f90, 258
- nuts.f90, 257
  - nuts, 258
- openwth.f90, 258
- operatn
  - operatn.f90, 259
- operatn.f90, 259
  - operatn, 259
- orgn
  - orgn.f90, 260
- orgn.f90, 259
  - orgn, 260
- orgncswat
  - orgncswat.f90, 260
- orgncswat.f90, 260
  - orgncswat, 260
- orgncswat2
  - orgncswat2.f90, 261
- orgncswat2.f90, 261
  - orgncswat2, 261
- origtile
  - origtile.f90, 262
- origtile.f90, 262
  - origtile, 262
- ovr\_sed
  - ovr\_sed.f90, 263
- ovr\_sed.f90, 262
  - ovr\_sed, 263
- oxygen\_saturation
  - oxygen\_saturation.f90, 263
- oxygen\_saturation.f90, 263
  - oxygen\_saturation, 263
- parm, 15
  - igropt, 99
- percmacro
  - percmacro.f90, 264
- percmacro.f90, 264
  - percmacro, 264
- percmain
  - percmain.f90, 265
- percmain.f90, 264
  - percmain, 265
- percmicro
  - percmicro.f90, 265
- percmicro.f90, 265
  - percmicro, 265
- pestlch
  - pestlch.f90, 267
- pestlch.f90, 267
  - pestlch, 267
- pestw.f90, 267
- pesty
  - pesty.f90, 268
- pesty.f90, 268
  - pesty, 268
- pgen
  - pgen.f90, 269
- pgen.f90, 268

- pgen, 269
- pgenhr.f90, 269
- pipe\_discharge
  - bmp\_wet\_pond.f90, 113
- pipeflow
  - pipeflow.f90, 270
- pipeflow.f90, 270
- pipeflow, 270
- pkq
  - pkq.f90, 271
- pkq.f90, 270
- pkq, 271
- plantmod
  - plantmod.f90, 272
- plantmod.f90, 271
- plantmod, 272
- plantop
  - plantop.f90, 272
- plantop.f90, 272
- plantop, 272
- pmeas
  - pmeas.f90, 273
- pmeas.f90, 273
- pmeas, 273
- pminrl
  - pminrl.f90, 274
- pminrl.f90, 273
- pminrl, 274
- pminrl2
  - pminrl2.f90, 274
- pminrl2.f90, 274
- pminrl2, 274
- pond
  - pond.f90, 275
- pond.f90, 275
- pond, 275
- pondhr
  - pondhr.f90, 276
- pondhr.f90, 275
- pondhr, 276
- pothole
  - pothole.f90, 276
- pothole.f90, 276
- pothole, 276
- print\_hyd
  - print\_hyd.f90, 278
- print\_hyd.f90, 278
- print\_hyd, 278
- psed
  - psed.f90, 279
- psed.f90, 278
- psed, 279
- qman
  - qman.f90, 279
- qman.f90, 279
- qman, 279
- rchaaf90, 280
- rchaaf90, 280
  - rchaaf90, 280
- rchday.f90, 281
- rchinit
  - rchinit.f90, 281
- rchinit.f90, 281
- rchinit, 281
- rchmon
  - rchmon.f90, 282
- rchmon.f90, 282
- rchmon, 282
- rchuse
  - rchuse.f90, 283
- rchuse.f90, 282
- rchuse, 283
- rchyr
  - rchyr.f90, 283
- rchyr.f90, 283
- rchyr, 283
- readatmodep.f90, 284
- readbsn.f90, 284
- readchm
  - readchm.f90, 285
- readchm.f90, 284
- readchm, 285
- readcnst
  - readcnst.f90, 285
- readcnst.f90, 285
- readcnst, 285
- readfcst.f90, 286
- readfert.f90, 286
- readfig.f90, 286
- readfile.f90, 287
- readgw
  - readgw.f90, 287
- readgw.f90, 287
- readgw, 287
- readhru
  - readhru.f90, 288
- readhru.f90, 288
- readhru, 288
- readinpt.f90, 288
- readlup.f90, 289
- readlwq
  - readlwq.f90, 289
- readlwq.f90, 289
- readlwq, 289
- readmgt
  - readmgt.f90, 290
- readmgt.f90, 290
- readmgt, 290
- readmon.f90, 290
- readops
  - readops.f90, 291
- readops.f90, 291
- readops, 291
- readpest.f90, 291

readplant.f90, 292  
readpnd  
    readpnd.f90, 292  
readpnd.f90, 292  
    readpnd, 292  
readres  
    readres.f90, 293  
readres.f90, 293  
    readres, 293  
readrte.f90, 293  
readru  
    readru.f90, 294  
readru.f90, 294  
    readru, 294  
readsdr  
    readsdr.f90, 295  
readsdr.f90, 294  
    readsdr, 295  
readsepticbz  
    readsepticbz.f90, 295  
readsepticbz.f90, 295  
    readsepticbz, 295  
readseptwq  
    readseptwq.f90, 297  
readseptwq.f90, 297  
    readseptwq, 297  
readsno  
    readsno.f90, 298  
readsno.f90, 297  
    readsno, 298  
readsol  
    readsol.f90, 298  
readsol.f90, 298  
    readsol, 298  
readsub  
    readsub.f90, 299  
readsub.f90, 299  
    readsub, 299  
readswq.f90, 299  
readtill.f90, 300  
readurban.f90, 300  
readwgn  
    readwgn.f90, 301  
readwgn.f90, 300  
    readwgn, 301  
readwus  
    readwus.f90, 301  
readwus.f90, 301  
    readwus, 301  
readwwq.f90, 302  
readyr  
    readyr.f90, 302  
readyr.f90, 302  
    readyr, 302  
reccnst  
    reccnst.f90, 303  
reccnst.f90, 303  
    reccnst, 303  
recday  
    recday.f90, 304  
recday.f90, 303  
    recday, 304  
rechour  
    rechour.f90, 304  
rechour.f90, 304  
    rechour, 304  
recmon  
    recmon.f90, 306  
recmon.f90, 306  
    recmon, 306  
recyear  
    recyear.f90, 307  
recyear.f90, 306  
    recyear, 307  
regres  
    regres.f90, 307  
regres.f90, 307  
    regres, 307  
res  
    res.f90, 308  
res.f90, 308  
    res, 308  
resetlu.f90, 309  
reshr  
    reshr.f90, 309  
reshr.f90, 309  
    reshr, 309  
resinit  
    resinit.f90, 310  
resinit.f90, 310  
    resinit, 310  
resnut  
    resnut.f90, 311  
resnut.f90, 310  
    resnut, 311  
rewind\_init.f90, 311  
rhgen.f90, 311  
rootfr  
    rootfr.f90, 312  
rootfr.f90, 312  
    rootfr, 312  
route  
    route.f90, 313  
route.f90, 312  
    route, 313  
routels.f90, 313  
routeunit  
    routeunit.f90, 314  
routeunit.f90, 313  
    routeunit, 314  
routes  
    routes.f90, 314  
routes.f90, 314  
    routes, 314  
rsedaa  
    rsedaa.f90, 315



rsedaa.f90, 315  
    rsedaa, 315  
rseday.f90, 315  
rsedmon  
    rsedmon.f90, 316  
rsedmon.f90, 316  
    rsedmon, 316  
rsedyr  
    rsedyr.f90, 317  
rsedyr.f90, 316  
    rsedyr, 317  
rtbact  
    rtbact.f90, 317  
rtbact.f90, 317  
    rtbact, 317  
rtday  
    rtday.f90, 318  
rtday.f90, 318  
    rtday, 318  
rteinit.f90, 318  
rthmusk  
    rthmusk.f90, 319  
rthmusk.f90, 319  
    rthmusk, 319  
rthpest  
    rthpest.f90, 320  
rthpest.f90, 319  
    rthpest, 320  
rthsed  
    rthsed.f90, 320  
rthsed.f90, 320  
    rthsed, 320  
rthvsc  
    rthvsc.f90, 321  
rthvsc.f90, 321  
    rthvsc, 321  
rtmusk  
    rtmusk.f90, 322  
rtmusk.f90, 321  
    rtmusk, 322  
rtout  
    rtout.f90, 322  
rtout.f90, 322  
    rtout, 322  
rtpest  
    rtpest.f90, 323  
rtpest.f90, 323  
    rtpest, 323  
rtsed  
    rtsed.f90, 324  
rtsed.f90, 323  
    rtsed, 324  
rtsed2  
    rtsed2.f90, 324  
rtsed2.f90, 324  
    rtsed2, 324  
sat\_excess  
    sat\_excess.f90, 325  
sat\_excess.f90, 325  
    sat\_excess, 325  
save  
    save.f90, 326  
save.f90, 326  
    save, 326  
saveconc  
    saveconc.f90, 327  
saveconc.f90, 326  
    saveconc, 327  
sched\_mgt  
    sched\_mgt.f90, 327  
sched\_mgt.f90, 327  
    sched\_mgt, 327  
schedule\_ops  
    schedule\_ops.f90, 328  
schedule\_ops.f90, 328  
    schedule\_ops, 328  
sim\_initday.f90, 328  
sim\_inityr.f90, 329  
simulate.f90, 329  
slrgen  
    slrgen.f90, 330  
slrgen.f90, 329  
    slrgen, 330  
smeas.f90, 330  
snom  
    snom.f90, 331  
snom.f90, 330  
    snom, 331  
soil\_chem  
    soil\_chem.f90, 331  
soil\_chem.f90, 331  
    soil\_chem, 331  
soil\_phys  
    soil\_phys.f90, 332  
soil\_phys.f90, 332  
    soil\_phys, 332  
soil\_write  
    soil\_write.f90, 333  
soil\_write.f90, 332  
    soil\_write, 333  
solp  
    solp.f90, 333  
solp.f90, 333  
    solp, 333  
solt.f90, 334  
std1.f90, 334  
std2.f90, 334  
std3.f90, 335  
stdaa.f90, 335  
storeinitial.f90, 335  
structure  
    structure.f90, 336  
structure.f90, 336  
    structure, 336  
sub\_subbasin  
    sub\_subbasin.f90, 337

sub\_subbasin.f90, 336  
     sub\_subbasin, 337  
 subaa  
     subaa.f90, 337  
 subaa.f90, 337  
     subaa, 337  
 subbasin  
     subbasin.f90, 338  
 subbasin.f90, 338  
     subbasin, 338  
 subday  
     subday.f90, 339  
 subday.f90, 338  
     subday, 339  
 submon.f90, 339  
 substor  
     substor.f90, 340  
 substor.f90, 339  
     substor, 340  
 subwq  
     subwq.f90, 340  
 subwq.f90, 340  
     subwq, 340  
 subyr.f90, 341  
 sumhyd.f90, 341  
 sumv  
     sumv.f90, 342  
 sumv.f90, 341  
     sumv, 342  
 surface  
     surface.f90, 342  
 surface.f90, 342  
     surface, 342  
 surfst\_h2o  
     surfst\_h2o.f90, 343  
 surfst\_h2o.f90, 343  
     surfst\_h2o, 343  
 surfstor  
     surfstor.f90, 344  
 surfstor.f90, 343  
     surfstor, 344  
 surq\_daycn  
     surq\_daycn.f90, 344  
 surq\_daycn.f90, 344  
     surq\_daycn, 344  
 surq\_greenampt  
     surq\_greenampt.f90, 345  
 surq\_greenampt.f90, 345  
     surq\_greenampt, 345  
 swbl  
     swbl.f90, 346  
 swbl.f90, 346  
     swbl, 346  
 sweep  
     sweep.f90, 347  
 sweep.f90, 346  
     sweep, 347  
 swu  
     swu.f90, 347  
 swu.f90, 347  
     swu, 347  
 tair  
     tair.f90, 348  
 tair.f90, 348  
     tair, 348  
 tgen  
     tgen.f90, 349  
 tgen.f90, 349  
     tgen, 349  
 theta  
     theta.f90, 350  
 theta.f90, 349  
     theta, 350  
 tillfactor  
     tillfactor.f90, 351  
 tillfactor.f90, 350  
     tillfactor, 351  
 tmeas.f90, 351  
 tran  
     tran.f90, 352  
 tran.f90, 352  
     tran, 352  
 transfer  
     transfer.f90, 353  
 transfer.f90, 352  
     transfer, 353  
 tstr  
     tstr.f90, 353  
 tstr.f90, 353  
     tstr, 353  
 ttcoef  
     ttcoef.f90, 354  
 ttcoef.f90, 354  
     ttcoef, 354  
 ttcoef\_wway.f90, 354  
 urb\_bmp  
     urb\_bmp.f90, 355  
 urb\_bmp.f90, 355  
     urb\_bmp, 355  
 urban  
     urban.f90, 356  
 urban.f90, 355  
     urban, 356  
 urbanhr  
     urbanhr.f90, 356  
 urbanhr.f90, 356  
     urbanhr, 356  
 varinit  
     varinit.f90, 357  
 varinit.f90, 357  
     varinit, 357  
 vbl  
     vbl.f90, 358  
 vbl.f90, 357

vbl, [358](#)  
virtual  
  virtual.f90, [359](#)  
virtual.f90, [359](#)  
  virtual, [359](#)  
volq  
  volq.f90, [360](#)  
volq.f90, [359](#)  
  volq, [360](#)  
  
washp  
  washp.f90, [360](#)  
washp.f90, [360](#)  
  washp, [360](#)  
watbal  
  watbal.f90, [361](#)  
watbal.f90, [361](#)  
  watbal, [361](#)  
water\_hru  
  water\_hru.f90, [362](#)  
water\_hru.f90, [361](#)  
  water\_hru, [362](#)  
watqual  
  watqual.f90, [362](#)  
watqual.f90, [362](#)  
  watqual, [362](#)  
watqual2  
  watqual2.f90, [363](#)  
watqual2.f90, [363](#)  
  watqual2, [363](#)  
wattable.f90, [363](#)  
watuse  
  watuse.f90, [364](#)  
watuse.f90, [364](#)  
  watuse, [364](#)  
weatgn  
  weatgn.f90, [365](#)  
weatgn.f90, [365](#)  
  weatgn, [365](#)  
wetlan  
  wetlan.f90, [366](#)  
wetlan.f90, [365](#)  
  wetlan, [366](#)  
wmeas.f90, [366](#)  
wndgen  
  wndgen.f90, [367](#)  
wndgen.f90, [366](#)  
  wndgen, [367](#)  
writea  
  writea.f90, [367](#)  
writea.f90, [367](#)  
  writea, [367](#)  
writeaa.f90, [368](#)  
writed.f90, [368](#)  
writem  
  writem.f90, [368](#)  
writem.f90, [368](#)  
  writem, [368](#)  
  
xmon.f90, [369](#)  
  
ysed  
  ysed.f90, [369](#)  
ysed.f90, [369](#)  
  ysed, [369](#)  
  
zero0.f90, [370](#)  
zero1.f90, [370](#)  
zero2.f90, [370](#)  
zero\_urban.f90, [371](#)  
zeroini.f90, [371](#)