```
//%REPLACE %CL% =("../CODELIB/CodeLibrary.mod")
//%GET %CL% standardInitialize()
/*
  2 compartment model for 2 species, A and B, both tracher (h) and
  non-tracer (c), with flow and competitive transporter
*/
math Comp2_hc_AB_Trans {
//%GET %CL% odeDomains()
// PARAMETERS
real Fp      = 1 ml/(g*min);           // Flow
real Ga2b    = 1 ml/(g*min); // Conversion rate of A to B
//%REPLACE (%pc%=("p","c"), %vol%=("0.05","0.60"), %flip%=("p2c","c2p") )
//%REPLACE %AB%=("A","B")
//%REPLACE %hc%=("h","c")
real V%pc% = %vol% ml/g;         // Volume of V%n%
real Surf    = 1 cm^2/g;
real SoV%pc% = Surf/V%pc%;
real Kd%hc%%AB%%pc% = 1 uM;
real kon%hc%%AB%%pc%      = 10 uM^(-1)*s^(-1);
real kof%hc%%AB%%pc%      = kon%hc%%AB%%pc%*Kd%hc%%AB%%pc%;
real Ttot    = 0.2 umol/cm^2;
real kT%flip%= 10 sec^(-1);
real kT%hc%%AB%%flip%     = 10 sec^(-1);
extern real %hc%%AB%in(t) mM;          // Inflow Conc
// DEPENDENT VARIABLES
real %hc%%AB%%pc%(t) mM;         // Conc
real %hc%%AB%out(t) mM;          // Outflow Conc
real %hc%%AB%%pc%ic = 0 mM;      // Init Conc
real T%pc%(t) umol/cm^2;
real T%hc%%AB%%pc%(t) umol/cm^2;
real T%hc%%AB%%pc%ic = 0 umol/cm^2;
// INITIAL CONDITIONS
when(t=t.min) %hc%%AB%%pc%=%hc%%AB%%pc%ic;
when(t=t.min) T%hc%%AB%%pc%=T%hc%%AB%%pc%ic;
when(t=t.min) T%pc% = Ttot/2;
// ORDINARY DIFFERENTIAL EQUATIONS
//%GET %CL% flowODECalc("C=%hc%%AB%p","F=Fp","V=Vp","Cin=%hc%%AB%in","Cout=%hc%%AB%out")
//%GET %CL% onOffMembraneCalc("M=%hc%%AB%%pc%","B=T%pc%","MB=T%hc%%AB%%pc%",
//%          "kon=kon%hc%%AB%%pc%","kof=kof%hc%%AB%%pc%","SoV=SoV%pc%")
//%GET %CL% flipa2bCalc("a=T%hc%%AB%p","b=T%hc%%AB%c",
//%                     "ka2b=kT%hc%%AB%p2c","kb2a=kT%hc%%AB%c2p")
//%GET %CL% flipa2bCalc("a=Tp","b=Tc",
//%                     "ka2b=kTp2c","kb2a=kTc2p")
//%GET %CL% reactionCalc1("A=%hc%Ac","B=%hc%Bc","V=Vc","G=Ga2b")
//%COLLECT("%hc%%AB%%pc%:t")
//%COLLECT("T%hc%%AB%%pc%:t")
//%COLLECT("T%pc%:t")
//%ENDREPLACE
//%ENDREPLACE
//%ENDREPLACE
}
//%GET %CL% standardClosing()
//%ENDREPLACE
```

```
//%GET randomPDF.mod randomPDFCheck()
//%GET CodeLibrary.mod curveStatJava()
import nsrunit; unit conversion on;

math RandomPerfusion {
real Nmax = 20;
//%REPLACE %n%=("#1#20")
//%REPLACE (%nm1%=("#1#19"),%np1%=("#2#20") )
real Fmean = 1 ml/(g*min);
//%GET randomPDF.mod randomFlows()
real F%n%=normf(%n%)*Fmean;
real weight%n%;
weight%n%=w(%n%);
real PSisf=1 ml/(g*min);
//%GET BTEX20.mod btex20ParmVar("C1=Cp%n%","C2=Cisf%n%","C1out=Cp%n%out",
//%             "F=F%n%","V1=Vp","V2=Visf")
//%GET BTEX20.mod btex20Calc("C1=Cp%n%","C2=Cisf%n%","C1out=Cp%n%out",
//%             "F=F%n%","V1=Vp","V2=Visf")
//%GET CodeLibrary.mod exchangeCalc("C1=Cisf%nm1%","C2=Cisf%np1%","PS=PSisf",
//%                                 "V1=Visf","V2=Visf")
real sumout(t) mM;
sumout = Cp%n%out*weight%n%*normf(%n%);
real Q(t) umol/(g*min);
when(t=t.min) Q=0;
Q=Fmean*(Cin-sumout);
//%COLLECT("sumout")
//%COLLECT("Cp%n%:t");
//%COLLECT("Cisf%n%:t");
//%ENDREPLACE
//%ENDREPLACE
//%GET CodeLibrary.mod transitCalc ("Cout=sumout")
}
```

```
math LaPlace2d { // Time dependent Laplacian PDE
//%REPLACE y%=("#0#20")
//%REPLACE (yi% =("#1#19"), yim1%=("#0#18"), yip1%=("#2#20"))

realDomain t ; t.min=0; t.max=1; t.delta=0.001;
realDomain x ; x.min=0; x.max=1; x.delta=1./20;
realDomain y ; y.min=0; y.max=1; y.delta=1./20.;
real D=1;
// VARIABLES
private real Cyy%(x,t);
//BOUNDARY NODES
real Bottom=1, Top=0, Left=1, Right=0;
              Cy0=Bottom* sin(PI*x);
              Cy20=Top*   sin(PI*x);
when(x=x.min) Cyyi%=Left*  sin(PI*yi%/20);
when(x=x.max) Cyyi%=Right*sin(PI*yi%/20);
//INITIAL CONDITIONS (interior nodes)
when(t=t.min) Cyyi% = if(x=x.min) Left*sin(PI*yi%/20) else if (x=x.max) Right*sin(PI*yi%/20)
else 0;
//PDEs
Cyyi%:t = D*((Cyyip1%+Cyyim1%-2*Cyyi%)/y.delta^2) + D*Cyyi%:x:x;
//ASSEMBLE ANSWERS
real C(x,y,t) =
      if( abs(y-y%/20) <y.delta/4) Cyy% else
        0;

//%ENDREPLACE
//%ENDREPLACE
}
```