```
%Main function: Calls each function of a PBPK model

clear all;                %clear all variables from previous simulations
close all;                %close all windows from previous simulations
clc;                      %clear command window


DefineParameters;         %model structure and parameters are defined
Population;               %the defined population is generated
Drug;                     %load drug files / in vitro-to-in vivo extrapolation
SolveODE;                 %solve the ordinary differential equations
PostProcessing;           %process the data from the ODE solution / output results
```

```matlab
function[] = DefineParameters()
%This function defines the population, model structure (PBPK compartments),
%the drugs, the virtual trial design and the simulation settings to be used

global DEF       %global DEF defines model parameters
global SYSTEM    %global SYSTEM defines sytem parameters
global DRUG      %global DRUG defines drug parameters
global STUDY     %global STUDY defines study design parameters
global MODEL     %global MODEL defines parameters important for modeling
global OBS       %global OBS saves observed parameters for the output

%define parameters used in the script
pop = 1;    com = 2;    seg = 3;    cyp = 4;    dru = 5;    pro = 6;    adm = 7;

%__Define Model parameters and the structure____________________________________
%%% Population %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DEF.AgingCaucasian = 1;    DEF.name{pop, DEF.AgingCaucasian} = 'AgingCaucasian';

%%% Compartments %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%RBC = red blood cells
DEF.lung      = 1;         DEF.name{com, DEF.lung}      = 'lung';
DEF.adipose   = 2;         DEF.name{com, DEF.adipose}   = 'adipose';
DEF.bone      = 3;         DEF.name{com, DEF.bone}      = 'bone';
DEF.brain     = 4;         DEF.name{com, DEF.brain}     = 'brain';
DEF.gonads    = 5;         DEF.name{com, DEF.gonads}    = 'gonads';
DEF.heart     = 6;         DEF.name{com, DEF.heart}     = 'heart';
DEF.kidney    = 7;         DEF.name{com, DEF.kidney}    = 'kidney';
DEF.muscle    = 8;         DEF.name{com, DEF.muscle}    = 'muscle';
DEF.skin      = 9;         DEF.name{com, DEF.skin}      = 'skin';
DEF.thymus    = 10;        DEF.name{com, DEF.thymus}    = 'thymus';
DEF.gut       = 11;        DEF.name{com, DEF.gut}       = 'gut';
DEF.spleen    = 12;        DEF.name{com, DEF.spleen}    = 'spleen';
DEF.pancreas  = 13;        DEF.name{com, DEF.pancreas}  = 'pancreas';
DEF.liver     = 14;        DEF.name{com, DEF.liver}     = 'liver';
DEF.lymphnode = 15;        DEF.name{com, DEF.lymphnode} = 'lymphnode';
DEF.remaining = 16;        DEF.name{com, DEF.remaining} = 'remaining';
DEF.plasma    = 17;        DEF.name{com, DEF.plasma}    = 'plasma';
DEF.RBC       = 18;        DEF.name{com, DEF.RBC}       = 'RBC';

Comp = [DEF.lung DEF.adipose DEF.bone DEF.brain DEF.gonads DEF.heart ...
        DEF.kidney DEF.muscle DEF.skin DEF.thymus DEF.gut DEF.spleen ...
        DEF.pancreas DEF.liver DEF.lymphnode DEF.remaining DEF.plasma ...
        DEF.RBC];
Org  = [DEF.lung DEF.adipose DEF.bone DEF.brain DEF.gonads DEF.heart ...
        DEF.kidney DEF.muscle DEF.skin DEF.thymus DEF.gut DEF.spleen ...
        DEF.pancreas DEF.liver DEF.lymphnode DEF.remaining];

%how many compartments are used with and without the blood?
CompNo = length(Comp);    SYSTEM.CompNo = CompNo;
OrgNo  = length(Org);     SYSTEM.OrgNo  = OrgNo;

%%% intestinal segments %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DEF.stomach  = 1;          DEF.name{seg, DEF.stomach}  = 'stomach';
DEF.duodenum = 2;           DEF.name{seg, DEF.duodenum} = 'duodenum';
DEF.jejunum  = 3;           DEF.name{seg, DEF.jejunum}  = 'jejunum';
DEF.ileum    = 4;           DEF.name{seg, DEF.ileum}    = 'ileum';
DEF.colon    = 5;           DEF.name{seg, DEF.colon}    = 'colon';
DEF.faeces   = 6;           DEF.name{seg, DEF.faeces}   = 'faeces';

Seg = [DEF.stomach DEF.duodenum DEF.jejunum DEF.ileum DEF.colon DEF.faeces];

%how many intestinal segments are modelled?
SegNo = length(Seg);    SYSTEM.SegNo = SegNo;

%%% CYP enzymes (for dynamic abundance for MBI and induction) %%%%%%%%%%%%%%%%%%
DEF.CYP2D6 = 1;             DEF.name{cyp, DEF.CYP2D6} = 'CYP2D6';
DEF.CYP3A4 = 2;             DEF.name{cyp, DEF.CYP3A4} = 'CYP3A4';
DEF.CYP3A5 = 3;             DEF.name{cyp, DEF.CYP3A5} = 'CYP3A5';
DEF.CYP2J2 = 4;             DEF.name{cyp, DEF.CYP2J2} = 'CYP2J2';

CYPli = [DEF.CYP2D6 DEF.CYP3A4 DEF.CYP3A5 DEF.CYP2J2];
CYPin = [DEF.CYP2D6 DEF.CYP3A4 DEF.CYP3A5];

%how many CYP enzymes are in the liver (li) or intestine (in)?
CYPliNo = length(CYPli);    SYSTEM.CYPliNo = CYPliNo;
CYPinNo = length(CYPin);    SYSTEM.CYPinNo = CYPinNo;

%%% Subcompartments %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%the number of subcompartments are also the number of equations

Sub(DEF.lung)       = 2;     Sub(DEF.adipose)    = 2;     Sub(DEF.bone)       = 2;
Sub(DEF.brain)      = 2;     Sub(DEF.gonads)     = 2;     Sub(DEF.heart)      = 2;
Sub(DEF.kidney)     = 2;     Sub(DEF.muscle)     = 2;     Sub(DEF.skin)       = 2;
Sub(DEF.thymus)     = 2;     Sub(DEF.gut)        = 1;     Sub(DEF.spleen)     = 2;
Sub(DEF.pancreas)   = 2;     Sub(DEF.liver)      = 2;     Sub(DEF.lymphnode) = 2;
Sub(DEF.remaining) = 2;     Sub(DEF.plasma)     = 0;     Sub(DEF.RBC)        = 0;

Sub(CompNo + DEF.stomach)  = 0;     Sub(CompNo + DEF.duodenum) = 2;
Sub(CompNo + DEF.jejunum)  = 2;     Sub(CompNo + DEF.ileum)    = 2;
Sub(CompNo + DEF.colon)    = 2;     Sub(CompNo + DEF.faeces)   = 0;

Sub(CompNo + SegNo + DEF.CYP2D6) = 3;     Sub(CompNo + SegNo + DEF.CYP3A4) = 3;
Sub(CompNo + SegNo + DEF.CYP3A5) = 3;     Sub(CompNo + SegNo + DEF.CYP2J2) = 0;

%define the number of subcompartments
SubNo = zeros(1, CompNo + SegNo + CYPliNo);
for tot = 1:(CompNo + SegNo + CYPliNo)
    if tot == 0
        SubNo(tot) = 0;

    else
        SubNo(tot) = sum(Sub(1:tot-1));
    end
end

SYSTEM.SubNo = SubNo;
```

```
%number of ODEs to solve
ODENo = CompNo + SegNo + CYPliNo + sum(Sub);    MODEL.ODENo = ODENo;

%%% Drugs %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DEF.darunavir   = 1;       DEF.name{dru, DEF.darunavir}   = 'darunavir';
DEF.ritonavir   = 2;       DEF.name{dru, DEF.ritonavir}   = 'ritonavir';
DEF.rivaroxaban = 3;       DEF.name{dru, DEF.rivaroxaban} = 'rivaroxaban';

DrugLib = [DEF.darunavir DEF.ritonavir DEF.rivaroxaban];

%How many drug files are in the libraray?
DEF.DrugLibNo = length(DrugLib);

%%% Main binding protein %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AAG = alpha acidic glycoprotein
DEF.albumin = 1;           DEF.name{pro, DEF.albumin} = 'albumin';
DEF.AAG     = 2;           DEF.name{pro, DEF.AAG}     = 'AAG';

%%% Route of administration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DEF.iv   = 1;              DEF.name{adm, DEF.iv}   = 'iv';
DEF.oral = 2;              DEF.name{adm, DEF.oral} = 'oral';

%==============================================================================
%__The user chooses the simulation settings_____________________________________
%%% Drug %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%how many drugs should be simulated in parallel?
DRUG.DrugNo = 3;

%enter the name of each simulated drug
DRUG.DrugName = zeros(1, DRUG.DrugNo);

for d = 1:DRUG.DrugNo
    switch d
        case 1    %drug 1
            DRUG.DrugName(d) = DEF.darunavir;

        case 2    %drug 2
            DRUG.DrugName(d) = DEF.ritonavir;

        case 3    %drug 3
            DRUG.DrugName(d) = DEF.rivaroxaban;

    end
end

%%% Virtual study design %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
STUDY.TrialNo    = 10;     %number of trials
STUDY.IndTrialNo = 12;     %number of individuals per trial

%total number of subjects to be simulated
STUDY.IndNo      = STUDY.TrialNo * STUDY.IndTrialNo;
STUDY.PropFem    = 0;     %proportion of women

STUDY.AgeMin     = 20;    %minimal age in the simulation in [years]
STUDY.AgeMax     = 50;    %maximum age in the simulation in [years]

STUDY.Resolution = 10;    %resolution of each time unit

%prepare empty vectors for dosing regimen
AdminRoute    = zeros(1, DRUG.DrugNo);     %route of administration
NoDoses       = zeros(1, DRUG.DrugNo);     %number of doses
Dose          = zeros(1, DRUG.DrugNo);     %dose in [mg]
StartDose     = zeros(1, DRUG.DrugNo);     %when is the drug given? [h]
DoseIntervall = zeros(1, DRUG.DrugNo);     %intervall between doses in [h]
LastTime      = zeros(1, DRUG.DrugNo);     %prolongation in [h]

for d = 1:DRUG.DrugNo
    switch d
        case 1
            AdminRoute(d)    = DEF.oral;
            NoDoses(d)       = 7;
            Dose(d)          = 800;
            StartDose(d)     = 0;
            DoseIntervall(d) = 24;
            LastTime(d)      = 0;

        case 2
            AdminRoute(d)    = DEF.oral;
            NoDoses(d)       = 7;
            Dose(d)          = 100;
            StartDose(d)     = 0;
            DoseIntervall(d) = 24;
            LastTime(d)      = 0;

        case 3
            AdminRoute(d)    = DEF.oral;
            NoDoses(d)       = 1;
            Dose(d)          = 10;
            StartDose(d)     = 144;
            DoseIntervall(d) = 24;
            LastTime(d)      = 24;
    end
end

%%% Enter observed data to compare to the simulated outcome %%%%%%%%%%%%%%%%%%%%
OBS.Time_Drug1 = [];
OBS.Conc_Drug1 = [];
OBS.SD_Drug1   = [];

OBS.Time_Drug2 = [];
OBS.Conc_Drug2 = [];
OBS.SD_Drug2    = [];

OBS.Time_Drug3 = [];
OBS.Conc_Drug3 = [];
OBS.SD_Drug3   = [];

OBS.Time_DDI1  = [];
OBS.Conc_DDI1  = [];
OBS.SD_DDI1    = [];

OBS.Time_DDI2  = [];
OBS.Conc_DDI2  = [];
OBS.SD_DDI2    = [];

OBS.Time_DDI3  = [];
OBS.Conc_DDI3  = [];
OBS.SD_DDI3    = [];

%================================================================================
%__set up a dose event matrix___________________________________________________
%define the name of columns for the dose event matrix
start = 1;    ende = 2;    dose = 3;    admin = 4;    res = 5;    mmr = 6;

NoColDoseEvent = length([start, ende, dose, admin, res, mmr]);
Regimen = zeros( max(NoDoses), NoColDoseEvent, DRUG.DrugNo);

%%% Combine all dosing events in one matrix %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%define the start and end of each dosing event for each drug
%dose and route of administration are assigned to each dosing event

for d = 1:DRUG.DrugNo
    for m = 1:NoDoses(d)

        %a case is defined for single doses
        if m == 1
            Regimen(1, start, d)  = 0 + StartDose(d);
            Regimen(1, ende, d)   = Regimen(1, start, d) + DoseIntervall(d) +...
                                    LastTime(d);

        %a case is defined for multiple doses
        else
            Regimen(1, start, d)  = 0 + StartDose(d);
            Regimen(1, ende, d) = Regimen(1, start, d) + DoseIntervall(d);

            Regimen(m, start, d)  = Regimen(m-1, ende, d);
            Regimen(m, ende, d) = Regimen(m, start, d) + DoseIntervall(d);

            %a case is defined for the last dose to prolong the elimination phase
            if m == NoDoses(d)
                Regimen(m, ende, d) = Regimen(m, start, d) +...
                                      DoseIntervall(d) +...
                                      LastTime(d);
            end
        end

        Regimen(m, dose, d)   = Dose(d);
        Regimen(m, admin, d) = AdminRoute(d);
        Regimen(m, res, d)   = (Regimen(m, ende, d) - Regimen(m, start, d)) .*...
                               STUDY.Resolution;

        if m == 1
            Regimen(1, mmr, d) = 0;

        else
            Regimen(m, mmr, d) = Regimen(m-1, mmr, d) +...
                                 (Regimen(m-1, ende, d) - Regimen(m-1, start, d)) .↙
*...
                                 STUDY.Resolution;
        end
    end
end

%%% find unique dosing events %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%extract dosing events for each drug and delete zero values
for d = 1:DRUG.DrugNo
    switch d
        case 1
            Drug1Regimen = sortrows(Regimen(:, :, d), start);
            Drug1Regimen( ~any(Drug1Regimen, 2), :) = [];

        case 2
            Drug2Regimen = sortrows(Regimen(:, :, d), start);
            Drug2Regimen( ~any(Drug2Regimen, 2), :) = [];

        case 3
            Drug3Regimen = sortrows(Regimen(:, :, d), start);
            Drug3Regimen( ~any(Drug3Regimen, 2), :) = [];
    end
end

%combine dosing regimens of the different drugs and look for unique events
switch DRUG.DrugNo
    case 1
        Drug1RegMat = Drug1Regimen;
        Drug1RegFun = RegimenFun(Drug1RegMat, NoDoses(1));

    case 2
        Drug1RegMat = [Drug1Regimen; Drug2Regimen];
        Drug1RegFun = RegimenFun(Drug1RegMat, NoDoses(1));

        Drug2RegMat = [Drug2Regimen; Drug1Regimen];
        Drug2RegFun = RegimenFun(Drug2RegMat, NoDoses(2));

    case 3
        Drug1RegMat = [Drug1Regimen; Drug2Regimen; Drug3Regimen];
        Drug1RegFun = RegimenFun(Drug1RegMat, NoDoses(1));

        Drug2RegMat = [Drug2Regimen; Drug1Regimen; Drug3Regimen];
        Drug2RegFun = RegimenFun(Drug2RegMat, NoDoses(2));
        Drug3RegMat = [Drug3Regimen; Drug1Regimen; Drug2Regimen];
        Drug3RegFun = RegimenFun(Drug3RegMat, NoDoses(3));
end

%prolongation of the terminal time needs to be considered for all drugs
switch DRUG.DrugNo
    case 1
        StartTime = Drug1RegFun(:, start);
        EndTime   = Drug1RegFun(:, ende);
    case 2
        StartTime = [Drug1RegFun(:, start); Drug2RegFun(:, start)];
        EndTime   = [Drug1RegFun(:, ende); Drug2RegFun(:, ende)];

    case 3
        StartTime = [Drug1RegFun(:, start); Drug2RegFun(:, start); Drug3RegFun(:, ↙
start)];
        EndTime   = [Drug1RegFun(:, ende); Drug2RegFun(:, ende); Drug3RegFun(:, ↙
ende)];
end

%find unique start and end time and calculate the resolution
UniqueStartT = unique(StartTime);
UniqueEndT   = unique(EndTime);
for d = 1:DRUG.DrugNo
    if NoDoses(d) ~= 0
        if LastTime(d) ~= 0
            UniqueEndT(end-1) = UniqueEndT(end);
            UniqueEndT(end) = [];
        end
    end
end
ResolutionEvent = (UniqueEndT - UniqueStartT) .* STUDY.Resolution;
ResolutionAll = zeros(length(UniqueStartT), 1);
for a = 1:length(UniqueStartT)
    if a == 1
        ResolutionAll(a) = 0;

    else
        ResolutionAll(a) = ResolutionAll(a-1) +...
                           (UniqueEndT(a-1) - UniqueStartT(a-1)) .* STUDY.↙
Resolution;
    end
end

%each drug becomes the same start / end time and resolution
switch DRUG.DrugNo
    case 1
        Drug1RegFun(:, start) = UniqueStartT;
        Drug1RegFun(:, ende)  = UniqueEndT;
        Drug1RegFun(:, res)   = ResolutionEvent;
        Drug1RegFun(:, mmr)   = ResolutionAll;
    case 2
        Drug1RegFun(:, start) = UniqueStartT;
        Drug1RegFun(:, ende)  = UniqueEndT;
        Drug1RegFun(:, res)   = ResolutionEvent;
        Drug1RegFun(:, mmr)   = ResolutionAll;

        Drug2RegFun(:, start) = UniqueStartT;
        Drug2RegFun(:, ende)  = UniqueEndT;
        Drug2RegFun(:, res)   = ResolutionEvent;
        Drug2RegFun(:, mmr)   = ResolutionAll;

    case 3
        Drug1RegFun(:, start) = UniqueStartT;
        Drug1RegFun(:, ende)  = UniqueEndT;
        Drug1RegFun(:, res)   = ResolutionEvent;
        Drug1RegFun(:, mmr)   = ResolutionAll;

        Drug2RegFun(:, start) = UniqueStartT;
        Drug2RegFun(:, ende)  = UniqueEndT;
        Drug2RegFun(:, res)   = ResolutionEvent;
        Drug2RegFun(:, mmr)   = ResolutionAll;

        Drug3RegFun(:, start) = UniqueStartT;
        Drug3RegFun(:, ende)  = UniqueEndT;
        Drug3RegFun(:, res)   = ResolutionEvent;
        Drug3RegFun(:, mmr)   = ResolutionAll;
end

%what is the number of total events in the simulation?
NoEvents = length(Drug1RegFun(:, start));    STUDY.NoEvents = NoEvents;

%comine the dosing regimen for all drugs
DoseEventMat = zeros(NoEvents, NoColDoseEvent, DRUG.DrugNo);
switch DRUG.DrugNo
    case 1
        DoseEventMat(:, :, 1) = Drug1RegFun;

    case 2
        DoseEventMat(:, :, 1) = Drug1RegFun;
        DoseEventMat(:, :, 2) = Drug2RegFun;

    case 3
        DoseEventMat(:, :, 1) = Drug1RegFun;
        DoseEventMat(:, :, 2) = Drug2RegFun;
        DoseEventMat(:, :, 3) = Drug3RegFun;
end

%save the dose event matrix globally
STUDY.DoseEventMat = DoseEventMat;
STUDY.Dose     = Dose;
STUDY.NoDoses = NoDoses;

%===============================================================================
%%% USED FUNCTION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%==============================================================================
function DrugReg = RegimenFun(DrugMat, NoDoses)

    %dosing events from other drugs are set to zero
    DrugMat(NoDoses + 1:end, [dose, admin, res, mmr]) = 0;

    %drug events are sorted based on the start time
    DrugSort = sortrows(DrugMat, start);

    %find unique time points
    [~, idx] = unique(DrugSort(:, start));

    %ouput results of the used function
    DrugReg = DrugSort(idx, :);
end

end
```

```
function[] = Population()
%This function generates the virtual population based on the user settings
%Attention: the normrnd command needs to Statistics and Machine Learning Toolbox

global DEF       %global DEF defines model parameters
global SYSTEM    %global SYSTEM defines sytem parameters
global STUDY     %global STUDY defines study design parameters

%__Age distribution______________________________________________________________
%Weibull distribution was fitted to data from Eurostat (Stader et al., 2018)
Age = round(61.73.*(-log(1 - rand(STUDY.IndNo, 1))).^(1/1.55));

%Weibull distribution are infinity and need to be truncated
for ind = 1:STUDY.IndNo
    while Age(ind) < STUDY.AgeMin || Age(ind) > STUDY.AgeMax
        Age(ind) = round(61.73.*(-log(1 - rand(1, 1))).^(1/1.55));
    end
end

%save age globally
SYSTEM.Age = Age;

%__Sex distribution______________________________________________________________
%number of women in the simulation - round ensure the number to be an integer
FemNo = round(STUDY.PropFem .* STUDY.IndNo);

%generate a matrix with random numbers
IndexSex = randperm(STUDY.IndNo);

%assign randomly a 1 to women and a 0 to men
Sex = zeros(STUDY.IndNo, 1);
Sex(IndexSex(1 : FemNo)) = 1;

%save sex globally
SYSTEM.Sex = Sex;

%__Anthropometric parameters_____________________________________________________
%Equations are published in Stader et al., 2018
%%% Body height in [cm] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Height_Mean = -0.0039.*Age.^2 + 0.238.*Age - 12.5.*Sex + 176;
Height_CV   = 3.8;
Height_Min  = [150, 140];
Height_Max  = [200, 180];

SYSTEM.Height = Calc_SysPar(Height_Mean, Height_CV, Height_Min, Height_Max);

%%% Body weight in [kg] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Weight_Mean = -0.0039.*Age.^2 + 1.12.*SYSTEM.Height + 0.611.*Age -...
              0.424.*Sex - 137;
Weight_CV   = 15.2;
Weight_Min  = [50, 40];
Weight_Max  = [110, 90];
Weight_NoVa = -0.0039.*Age.^2 + 1.12.*Height_Mean + 0.611.*Age - 0.424.*Sex - 137;
```

```
SYSTEM.Weight = Calc_SysPar(Weight_Mean, Weight_CV, Weight_Min, Weight_Max);

%%% Body Surface Area (BSA) according to DuBois & DuBois %%%%%%%%%%%%%%%%%%%%%%
SYSTEM.BSA = 0.007184 .* SYSTEM.Height.^0.725 .* SYSTEM.Weight.^0.425;
BSA_NoVa   = 0.007184 .* Height_Mean.^0.725 .* Weight_NoVa.^0.425;

%__CYP enzymes phenotypes__________________________________________________________
%define variables
em = 1;    %EM = enhanced metaboliseres (2 allels)    - assign a random 1
pm = 2;    %PM = poor metaboliseres (0 allels)        - assign a random 0
um = 3;    %UM = ultrarapid metaboliseres (4 allels)  - assign a random 2

%we set up the frequencies for CYP enzymes
FreqCYP = zeros(SYSTEM.CYPliNo, length([em, pm, um]));

%enter the frequencies when available
FreqCYP(DEF.CYP2D6, pm) = 0.08;
FreqCYP(DEF.CYP2D6, um) = 0.05;
FreqCYP(DEF.CYP3A5, pm) = 0.83;

%how many subjects per phenotype?
PhenNoCYP = round(FreqCYP .* STUDY.IndNo);
PhenNoCYP(:, em) = STUDY.IndNo - PhenNoCYP(:, pm) - PhenNoCYP(:, um);

%prepare vector for loop
PheCYP  = ones(STUDY.IndNo, SYSTEM.CYPliNo);
CYP_EM  = num2cell(zeros(SYSTEM.CYPliNo), 1);
CYP_PM  = num2cell(zeros(SYSTEM.CYPliNo), 1);
CYP_UM  = num2cell(zeros(SYSTEM.CYPliNo), 1);
CYP_All = num2cell(zeros(SYSTEM.CYPliNo), 1);

for cyp = 1:SYSTEM.CYPliNo
    CYP_EM{cyp}  = ones(PhenNoCYP(cyp, em), 1);      %EMs get a 1
    CYP_PM{cyp}  = zeros(PhenNoCYP(cyp, pm), 1);     %PMs get a 0
    CYP_UM{cyp}  = 2*ones(PhenNoCYP(cyp, um), 1);    %UMs get a 3

    %combine all phenotypes
    CYP_All{cyp} = [CYP_EM{cyp}; CYP_PM{cyp}; CYP_UM{cyp}];

    %randomly assign 0 (PM) / 1 (EM) / 2 (UM) based on frequencies
    PheCYP(:, cyp) = CYP_All {cyp} (randperm(length(CYP_All {cyp} (:))));
end

%__Blood parameters________________________________________________________________
%all parameters are from Stader et al., 2018

%%% haematocrit (HCT) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
HCT_Mean = 0.443 - 0.033.*Sex;
HCT_CV   = 14.4;
HCT_Min  = [0.3, 0.3];
HCT_Max  = [0.5, 0.5];
```

```
SYSTEM.HCT = Calc_SysPar(HCT_Mean, HCT_CV, HCT_Min, HCT_Max);

%%% albumin (HSA) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Albumin_Mean = - 0.0709.*Age + 47.7;
Albumin_CV   = 7.9;
Albumin_Min  = [35.8, 35.8];
Albumin_Max  = [50.2, 50.2];

SYSTEM.Albumin = Calc_SysPar(Albumin_Mean, Albumin_CV, Albumin_Min, Albumin_Max);

%%% alpha-acidic glycoprotein (AAG) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
AAG_Mean = 0.798*ones(STUDY.IndNo, 1);
AAG_CV   = 24.3;
AAG_Min  = [0.476, 0.476];
AAG_Max  = [1.22, 1.22];

SYSTEM.AAG = Calc_SysPar(AAG_Mean, AAG_CV, AAG_Min, AAG_Max);

%__Organ weights (Worg) in [kg]_________________________________________________
%all parameters are from Stader et al., 2018

SYSTEM.Worg = zeros(STUDY.IndNo, SYSTEM.CompNo);

%%% Lungs (LU) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WLU_Mean = exp(0.00771.*Age + 0.0279.*SYSTEM.Height - 5.58);
WLU_CV   = 0;
WLU_Min  = [0.00453.*SYSTEM.Weight, 0.00453.*SYSTEM.Weight,];
WLU_Max  = [0.0122.*SYSTEM.Weight, 0.0122.*SYSTEM.Weight,];
WLU_NoVa = exp(0.00771.*Age + 0.0279.*Height_Mean - 5.58);

SYSTEM.Worg(:, DEF.lung) = Calc_SysPar(WLU_Mean, WLU_CV, WLU_Min, WLU_Max);

%%% Adipose (AD) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WAD_Mean = abs(0.68.*SYSTEM.Weight - 0.56.*SYSTEM.Height + 6.1.*Sex + 65);
WAD_CV   = 29.6;
WAD_Min  = [0.10.*SYSTEM.Weight, 0.21.*SYSTEM.Weight];
WAD_Max  = [0.42.*SYSTEM.Weight, 0.59.*SYSTEM.Weight];
WAD_NoVa = abs(0.68.*Weight_NoVa - 0.56.*Height_Mean + 6.1.*Sex + 65);

SYSTEM.Worg(:, DEF.adipose) = Calc_SysPar(WAD_Mean, WAD_CV, WAD_Min, WAD_Max);

%%% Bone (BO) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WBO_Mean = exp(0.024.*SYSTEM.Height - 1.9);
WBO_CV   = 13.2;
WBO_Min  = [0.079.*SYSTEM.Weight, 0.079.*SYSTEM.Weight];
WBO_Max  = [0.13.*SYSTEM.Weight, 0.13.*SYSTEM.Weight];
WBO_NoVa = exp(0.024.*Height_Mean - 1.9);

SYSTEM.Worg(:, DEF.bone) = Calc_SysPar(WBO_Mean, WBO_CV, WBO_Min, WBO_Max);

%%% Brain (BR) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WBR_Mean = exp(-0.00075.*Age + 0.00778.*SYSTEM.Height - 0.97);
WBR_CV   = 9.0;
```

```
WBR_Min  = [0.0168.*SYSTEM.Weight, 0.0168.*SYSTEM.Weight];
WBR_Max  = [0.0295.*SYSTEM.Weight, 0.0295.*SYSTEM.Weight];
WBR_NoVa = exp(-0.00075.*Age + 0.00778.*Height_Mean - 0.97);

SYSTEM.Worg(:, DEF.brain) = Calc_SysPar(WBR_Mean, WBR_CV, WBR_Min, WBR_Max);

%%% Gonads (GO) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WGO_Mean = -0.00022.*Age - 0.00034.*SYSTEM.Weight - 0.030.*Sex + 0.082;
WGO_CV   = 34.8;
WGO_Min  = [0.00032.*SYSTEM.Weight, 0.00008.*SYSTEM.Weight];
WGO_Max  = [0.00066.*SYSTEM.Weight, 0.00017.*SYSTEM.Weight];
WGO_NoVa = -0.00022.*Age - 0.00034.*Weight_NoVa - 0.030.*Sex + 0.082;

SYSTEM.Worg(:, DEF.gonads) = Calc_SysPar(WGO_Mean, WGO_CV, WGO_Min, WGO_Max);

%%% Heart (HE) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WHE_Mean = 0.34.*SYSTEM.BSA + 0.0018.*Age - 0.36;
WHE_CV   = 19.9;
WHE_Min  = [0.0018.*SYSTEM.Weight, 0.0018.*SYSTEM.Weight];
WHE_Max  = [0.0076.*SYSTEM.Weight, 0.0076.*SYSTEM.Weight];
WHE_NoVa = 0.34.*BSA_NoVa + 0.0018.*Age - 0.36;

SYSTEM.Worg(:, DEF.heart) = Calc_SysPar(WHE_Mean, WHE_CV, WHE_Min, WHE_Max);

%%% Kidney (KI) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WKI_Mean = -0.00038.*Age - 0.056.*Sex + 0.33;
WKI_CV   = 21.2;
WKI_Min  = [0, 0];
WKI_Max  = [1, 1];
WKI_NoVa = -0.00038.*Age - 0.056.*Sex + 0.33;

SYSTEM.Worg(:, DEF.kidney) = Calc_SysPar(WKI_Mean, WKI_CV, WKI_Min, WKI_Max);

%%% Muscle (MU) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WMU_Mean = 17.9.*SYSTEM.BSA - 0.0667.*Age - 5.68.*Sex - 1.22;
WMU_CV   = 11.8;
WMU_Min  = [0.310.*SYSTEM.Weight, 0.199.*SYSTEM.Weight];
WMU_Max  = [0.459.*SYSTEM.Weight, 0.388.*SYSTEM.Weight];
WMU_NoVa = 17.9.*BSA_NoVa - 0.0667.*Age - 5.68.*Sex - 1.22;

SYSTEM.Worg(:, DEF.muscle) = Calc_SysPar(WMU_Mean, WMU_CV, WMU_Min, WMU_Max);

%%% Skin (SK) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WSK_Mean = exp(-0.0058.*Age - 0.37.*Sex + 1.13);
WSK_CV   = 8.3;
WSK_Min  = [0.0094.*SYSTEM.Weight, 0.0054.*SYSTEM.Weight];
WSK_Max  = [0.0479.*SYSTEM.Weight, 0.0411.*SYSTEM.Weight];
WSK_NoVa = exp(-0.0058.*Age - 0.37.*Sex + 1.13);

SYSTEM.Worg(:, DEF.skin) = Calc_SysPar(WSK_Mean, WSK_CV, WSK_Min, WSK_Max);

%%% Thymus (TH) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WTH_Mean = 0.0221*ones(STUDY.IndNo, 1);
```

```
WTH_CV   = 44.8;
WTH_Min  = [0.00016.*SYSTEM.Weight, 0.00016.*SYSTEM.Weight];
WTH_Max  = [0.00046.*SYSTEM.Weight, 0.00046.*SYSTEM.Weight];
WTH_NoVa = 0.0221*ones(STUDY.IndNo, 1);

SYSTEM.Worg(:, DEF.thymus) = Calc_SysPar(WTH_Mean, WTH_CV, WTH_Min, WTH_Max);

%%% Gut (GU) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WGU_Mean = 3E-06.*SYSTEM.Height.^2.49;
WGU_CV   = 7.3;
WGU_Min  = [0, 0];
WGU_Max  = [2, 2];
WGU_NoVa = 3E-06.*Height_Mean.^2.49;

SYSTEM.Worg(:, DEF.gut) = Calc_SysPar(WGU_Mean, WGU_CV, WGU_Min, WGU_Max);

%%% Spleen (SP) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WSP_Mean = exp(1.13.*SYSTEM.BSA - 3.93);
WSP_CV   = 51.7;
WSP_Min  = [0.00098.*SYSTEM.Weight, 0.00098.*SYSTEM.Weight];
WSP_Max  = [0.00321.*SYSTEM.Weight, 0.00321.*SYSTEM.Weight];
WSP_NoVa = exp(1.13.*BSA_NoVa - 3.93);

SYSTEM.Worg(:, DEF.spleen) = Calc_SysPar(WSP_Mean, WSP_CV, WSP_Min, WSP_Max);

%%% Pancreas (PA) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WPA_Mean = 0.103*ones(STUDY.IndNo, 1);
WPA_CV   = 27.8;
WPA_Min  = [0, 0];
WPA_Max  = [1, 1];
WPA_NoVa = 0.103*ones(STUDY.IndNo, 1);

SYSTEM.Worg(:, DEF.pancreas) = Calc_SysPar(WPA_Mean, WPA_CV, WPA_Min, WPA_Max);

%%% Liver (LI) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WLI_Mean = exp(0.87.*SYSTEM.BSA - 0.014.*Age - 1.0);
WLI_CV   = 23.7;
WLI_Min  = [0.0147.*SYSTEM.Weight, 0.0147.*SYSTEM.Weight];
WLI_Max  = [0.0332.*SYSTEM.Weight, 0.0332.*SYSTEM.Weight];
WLI_NoVa = exp(0.87.*BSA_NoVa - 0.014.*Age - 1.0);

SYSTEM.Worg(:, DEF.liver) = Calc_SysPar(WLI_Mean, WLI_CV, WLI_Min, WLI_Max);

%%% Lymphnode (LN) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%data are from Gill et al., 2016
WLN_Mean = 0.00386.*SYSTEM.Weight;
WLN_CV   = 30;
WLN_Min  = [0, 0];
WLN_Max  = [1, 1];
WLN_NoVa = 0.00386.*Weight_NoVa;

SYSTEM.Worg(:, DEF.lymphnode) = Calc_SysPar(WLN_Mean, WLN_CV, WLN_Min, WLN_Max);
```

```
%%% Blood %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%weight of the entire blood
WBL = -0.0098.*Age - 1.89.*Sex + 6.06;

WorgBlood = normrnd(WBL, (10.4/100).*WBL);

%blood weight is split into plasma and red blood cells
SYSTEM.Worg(:, DEF.plasma) = WorgBlood.*(1 - 0.91.*SYSTEM.HCT);
SYSTEM.Worg(:, DEF.RBC)    = WorgBlood - SYSTEM.Worg(:, DEF.plasma);

%weight of venous (VB) and arterial blood (AB)
SYSTEM.Wvein   = (2/3).*WorgBlood;
SYSTEM.Wartery = (1/3).*WorgBlood;

%%% Remaining (RE) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SYSTEM.Worg(:, DEF.remaining) = SYSTEM.Weight - ...
                               SYSTEM.Worg(:, DEF.lung)      -...
                               SYSTEM.Worg(:, DEF.adipose)   -...
                               SYSTEM.Worg(:, DEF.bone)      -...
                               SYSTEM.Worg(:, DEF.brain)     -...
                               SYSTEM.Worg(:, DEF.gonads)    -...
                               SYSTEM.Worg(:, DEF.heart)     -...
                               SYSTEM.Worg(:, DEF.kidney)    -...
                               SYSTEM.Worg(:, DEF.muscle)    -...
                               SYSTEM.Worg(:, DEF.skin)      -...
                               SYSTEM.Worg(:, DEF.thymus)    -...
                               SYSTEM.Worg(:, DEF.gut)       -...
                               SYSTEM.Worg(:, DEF.spleen)    -...
                               SYSTEM.Worg(:, DEF.pancreas)  -...
                               SYSTEM.Worg(:, DEF.liver)     -...
                               SYSTEM.Worg(:, DEF.lymphnode) -...
                               SYSTEM.Worg(:, DEF.plasma)    -...
                               SYSTEM.Worg(:, DEF.RBC);

%because of the random variability, the remaining organ weight can be negative
%in these cases, the additional random variability is removed
for ind = 1:STUDY.IndNo
    if SYSTEM.Worg(ind, DEF.remaining) < 0
        SYSTEM.Height(ind) = Height_Mean(ind);
        SYSTEM.Weight(ind) = Weight_NoVa(ind);
        SYSTEM.BSA(ind)    = BSA_NoVa(ind);

        SYSTEM.Worg(ind, DEF.lung)      = WLU_NoVa(ind);
        SYSTEM.Worg(ind, DEF.adipose)   = WAD_NoVa(ind);
        SYSTEM.Worg(ind, DEF.bone)      = WBO_NoVa(ind);
        SYSTEM.Worg(ind, DEF.brain)     = WBR_NoVa(ind);
        SYSTEM.Worg(ind, DEF.gonads)    = WGO_NoVa(ind);
        SYSTEM.Worg(ind, DEF.heart)     = WHE_NoVa(ind);
        SYSTEM.Worg(ind, DEF.kidney)    = WKI_NoVa(ind);
        SYSTEM.Worg(ind, DEF.muscle)    = WMU_NoVa(ind);
        SYSTEM.Worg(ind, DEF.skin)      = WSK_NoVa(ind);
        SYSTEM.Worg(ind, DEF.thymus)    = WTH_NoVa(ind);
        SYSTEM.Worg(ind, DEF.gut)       = WGU_NoVa(ind);
        SYSTEM.Worg(ind, DEF.spleen)    = WSP_NoVa(ind);
        SYSTEM.Worg(ind, DEF.pancreas)  = WPA_NoVa(ind);
        SYSTEM.Worg(ind, DEF.liver)     = WLI_NoVa(ind);
        SYSTEM.Worg(ind, DEF.lymphnode) = WLN_NoVa(ind);
        SYSTEM.Worg(ind, DEF.plasma)    = WBL(ind) * (1 - 0.91.*HCT_Mean(ind));
        SYSTEM.Worg(ind, DEF.RBC)       = WBL(ind) - SYSTEM.Worg(ind, DEF.plasma);
    end
end

SYSTEM.Worg(:, DEF.remaining) = SYSTEM.Weight - ...
                               SYSTEM.Worg(:, DEF.lung)      -...
                               SYSTEM.Worg(:, DEF.adipose)   -...
                               SYSTEM.Worg(:, DEF.bone)      -...
                               SYSTEM.Worg(:, DEF.brain)     -...
                               SYSTEM.Worg(:, DEF.gonads)    -...
                               SYSTEM.Worg(:, DEF.heart)     -...
                               SYSTEM.Worg(:, DEF.kidney)    -...
                               SYSTEM.Worg(:, DEF.muscle)    -...
                               SYSTEM.Worg(:, DEF.skin)      -...
                               SYSTEM.Worg(:, DEF.thymus)    -...
                               SYSTEM.Worg(:, DEF.gut)       -...
                               SYSTEM.Worg(:, DEF.spleen)    -...
                               SYSTEM.Worg(:, DEF.pancreas)  -...
                               SYSTEM.Worg(:, DEF.liver)     -...
                               SYSTEM.Worg(:, DEF.lymphnode) -...
                               SYSTEM.Worg(:, DEF.plasma)    -...
                               SYSTEM.Worg(:, DEF.RBC);

%__Organ density (OrgDen) [kg/L]_______________________________________________
%Organ densities are from the ICRP 1975 (Snyder) and 2002 (Valentin)
SYSTEM.OrgDen = ones(1, SYSTEM.CompNo);

SYSTEM.OrgDen(DEF.lung)       = 1.0;      %free of blood and air
SYSTEM.OrgDen(DEF.adipose)    = 0.916;
SYSTEM.OrgDen(DEF.bone)       = 1.9;
SYSTEM.OrgDen(DEF.brain)      = 1.04;
SYSTEM.OrgDen(DEF.gonads)     = 1.045;    %combined from males and females
SYSTEM.OrgDen(DEF.heart)      = 1.03;
SYSTEM.OrgDen(DEF.kidney)     = 1.05;
SYSTEM.OrgDen(DEF.muscle)     = 1.041;
SYSTEM.OrgDen(DEF.skin)       = 1.1;
SYSTEM.OrgDen(DEF.thymus)     = 1.025;
SYSTEM.OrgDen(DEF.gut)        = 1.042;
SYSTEM.OrgDen(DEF.spleen)     = 1.06;
SYSTEM.OrgDen(DEF.pancreas)   = 1.045;
SYSTEM.OrgDen(DEF.liver)      = 1.08;     %Heineman et al. (1999)
SYSTEM.OrgDen(DEF.lymphnode) = 1.0;       %no data available; assume 1.0
SYSTEM.OrgDen(DEF.plasma)     = 1.027;
SYSTEM.OrgDen(DEF.RBC)        = 1.09;

%use the same organ density for each subject
SYSTEM.OrgDen = repmat(SYSTEM.OrgDen, STUDY.IndNo, 1);
```

```
%use the weighted mean of all used tissues for the remaining organ
SYSTEM.OrgDen(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.OrgDen, 2)./...
                                  sum(SYSTEM.Worg, 2);

%__Organ volume (Vorg) [L]______________________________________________________
SYSTEM.Vorg = SYSTEM.Worg./SYSTEM.OrgDen;

SYSTEM.Vvein    = SYSTEM.Wvein./1.06;
SYSTEM.Vartery = SYSTEM.Wartery./1.06;

%__Tissue composition___________________________________________________________
%Values published by Gill et al., 2016 and Jamei et al., 2009 are used
%thymus data are from rat (Rodgers & Rowland, 2005)
%gonad data are from Pierson et al., 1978, Bieri & Privali, 1965 and Diagne et al.,↙
1983
%lymphnode data are from Zhu et al., 1996

SYSTEM.FraEW = ones(1, SYSTEM.CompNo);    %fraction of extracellular water
SYSTEM.FraIW = ones(1, SYSTEM.CompNo);    %fraction of intracellular water
SYSTEM.FraNL = ones(1, SYSTEM.CompNo);    %fraction of neutral lipids
SYSTEM.FraNP = ones(1, SYSTEM.CompNo);    %fraction of phospholipids
SYSTEM.AP    = ones(1, SYSTEM.CompNo);    %acidic phospholipids in [mg/g]
SYSTEM.KpHSA = ones(1, SYSTEM.CompNo);    %partition coefficient of albumin

SYSTEM.FraEW(DEF.lung)       = 0.348;     SYSTEM.FraIW(DEF.lung)       = 0.463;
SYSTEM.FraEW(DEF.adipose)    = 0.141;     SYSTEM.FraIW(DEF.adipose)    = 0.039;
SYSTEM.FraEW(DEF.bone)       = 0.098;     SYSTEM.FraIW(DEF.bone)       = 0.341;
SYSTEM.FraEW(DEF.brain)      = 0.092;     SYSTEM.FraIW(DEF.brain)      = 0.678;
SYSTEM.FraEW(DEF.gonads)     = 0.239;     SYSTEM.FraIW(DEF.gonads)     = 0.561;
SYSTEM.FraEW(DEF.heart)      = 0.313;     SYSTEM.FraIW(DEF.heart)      = 0.445;
SYSTEM.FraEW(DEF.kidney)     = 0.283;     SYSTEM.FraIW(DEF.kidney)     = 0.5;
SYSTEM.FraEW(DEF.muscle)     = 0.091;     SYSTEM.FraIW(DEF.muscle)     = 0.669;
SYSTEM.FraEW(DEF.skin)       = 0.623;     SYSTEM.FraIW(DEF.skin)       = 0.0947;
SYSTEM.FraEW(DEF.thymus)     = 0.150;     SYSTEM.FraIW(DEF.thymus)     = 0.626;
SYSTEM.FraEW(DEF.gut)        = 0.267;     SYSTEM.FraIW(DEF.gut)        = 0.451;
SYSTEM.FraEW(DEF.spleen)     = 0.208;     SYSTEM.FraIW(DEF.spleen)     = 0.58;
SYSTEM.FraEW(DEF.pancreas)   = 0.12;      SYSTEM.FraIW(DEF.pancreas)   = 0.664;
SYSTEM.FraEW(DEF.liver)      = 0.165;     SYSTEM.FraIW(DEF.liver)      = 0.586;
SYSTEM.FraEW(DEF.lymphnode) = 0.208;      SYSTEM.FraIW(DEF.lymphnode) = 0.58;
SYSTEM.FraEW(DEF.plasma)     = 0.945;     SYSTEM.FraIW(DEF.plasma)     = 0;
SYSTEM.FraEW(DEF.RBC)        = 0;         SYSTEM.FraIW(DEF.RBC)        = 0.666;

SYSTEM.FraNL(DEF.lung)       = 0.003;     SYSTEM.FraNP(DEF.lung)       = 0.009;
SYSTEM.FraNL(DEF.adipose)    = 0.79;      SYSTEM.FraNP(DEF.adipose)    = 0.002;
SYSTEM.FraNL(DEF.bone)       = 0.074;     SYSTEM.FraNP(DEF.bone)       = 0.0011;
SYSTEM.FraNL(DEF.brain)      = 0.051;     SYSTEM.FraNP(DEF.brain)      = 0.0565;
SYSTEM.FraNL(DEF.gonads)     = 0.007;     SYSTEM.FraNP(DEF.gonads)     = 0.0077;
SYSTEM.FraNL(DEF.heart)      = 0.015;     SYSTEM.FraNP(DEF.heart)      = 0.0166;
SYSTEM.FraNL(DEF.kidney)     = 0.0207;    SYSTEM.FraNP(DEF.kidney)     = 0.0162;
SYSTEM.FraNL(DEF.muscle)     = 0.0238;    SYSTEM.FraNP(DEF.muscle)     = 0.0072;
SYSTEM.FraNL(DEF.skin)       = 0.0248;    SYSTEM.FraNP(DEF.skin)       = 0.0111;
SYSTEM.FraNL(DEF.thymus)     = 0.017;     SYSTEM.FraNP(DEF.thymus)     = 0.0092;
SYSTEM.FraNL(DEF.gut)        = 0.0487;    SYSTEM.FraNP(DEF.gut)        = 0.0163;
SYSTEM.FraNL(DEF.spleen)    = 0.0201;    SYSTEM.FraNP(DEF.spleen)    = 0.0198;
SYSTEM.FraNL(DEF.pancreas)  = 0.041;     SYSTEM.FraNP(DEF.pancreas)  = 0.0093;
SYSTEM.FraNL(DEF.liver)     = 0.0348;    SYSTEM.FraNP(DEF.liver)     = 0.0252;
SYSTEM.FraNL(DEF.lymphnode) = 0.0201;    SYSTEM.FraNP(DEF.lymphnode) = 0.0198;
SYSTEM.FraNL(DEF.plasma)    = 0.35;      SYSTEM.FraNP(DEF.plasma)    = 0.225;
SYSTEM.FraNL(DEF.RBC)       = 0.17;      SYSTEM.FraNP(DEF.RBC)       = 0.29;

SYSTEM.AP(DEF.lung)         = 0.5;       SYSTEM.KpHSA(DEF.lung)      = 0.212;
SYSTEM.AP(DEF.adipose)      = 0.4;       SYSTEM.KpHSA(DEF.adipose)   = 0.021;
SYSTEM.AP(DEF.bone)         = 0.67;      SYSTEM.KpHSA(DEF.bone)      = 0.1;
SYSTEM.AP(DEF.brain)        = 0.4;       SYSTEM.KpHSA(DEF.brain)     = 0.048;
SYSTEM.AP(DEF.gonads)       = 1.23;      SYSTEM.KpHSA(DEF.gonads)    = 0.048;
SYSTEM.AP(DEF.heart)        = 3.07;      SYSTEM.KpHSA(DEF.heart)     = 0.157;
SYSTEM.AP(DEF.kidney)       = 2.48;      SYSTEM.KpHSA(DEF.kidney)    = 0.13;
SYSTEM.AP(DEF.muscle)       = 2.49;      SYSTEM.KpHSA(DEF.muscle)    = 0.025;
SYSTEM.AP(DEF.skin)         = 1.32;      SYSTEM.KpHSA(DEF.skin)      = 0.277;
SYSTEM.AP(DEF.thymus)       = 2.3;       SYSTEM.KpHSA(DEF.thymus)    = 0.075;
SYSTEM.AP(DEF.gut)          = 2.84;      SYSTEM.KpHSA(DEF.gut)       = 0.158;
SYSTEM.AP(DEF.spleen)       = 2.81;      SYSTEM.KpHSA(DEF.spleen)    = 0.097;
SYSTEM.AP(DEF.pancreas)     = 1.67;      SYSTEM.KpHSA(DEF.pancreas)  = 0.06;
SYSTEM.AP(DEF.liver)        = 5.09;      SYSTEM.KpHSA(DEF.liver)     = 0.086;
SYSTEM.AP(DEF.lymphnode)    = 2.81;      SYSTEM.KpHSA(DEF.lymphnode) = 0.097;
SYSTEM.AP(DEF.plasma)       = 0.04;      SYSTEM.KpHSA(DEF.plasma)    = 1.0;
SYSTEM.AP(DEF.RBC)          = 0.44;      SYSTEM.KpHSA(DEF.RBC)       = 0.0;
%KpALB does not change with age (Yan et al., 1968)

%there is no variability of tissue composition parameters
SYSTEM.FraEW = repmat(SYSTEM.FraEW, STUDY.IndNo, 1);
SYSTEM.FraIW = repmat(SYSTEM.FraIW, STUDY.IndNo, 1);
SYSTEM.FraNL = repmat(SYSTEM.FraNL, STUDY.IndNo, 1);
SYSTEM.FraNP = repmat(SYSTEM.FraNP, STUDY.IndNo, 1);
SYSTEM.AP    = repmat(SYSTEM.AP, STUDY.IndNo, 1);
SYSTEM.KpHSA = repmat(SYSTEM.KpHSA, STUDY.IndNo, 1);

%remaining organ will always be the weighted mean
SYSTEM.FraEW(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.FraEW, 2)./sum(SYSTEM. ↙
Worg, 2);
SYSTEM.FraIW(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.FraIW, 2)./sum(SYSTEM. ↙
Worg, 2);
SYSTEM.FraNL(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.FraNL, 2)./sum(SYSTEM. ↙
Worg, 2);
SYSTEM.FraNP(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.FraNP, 2)./sum(SYSTEM. ↙
Worg, 2);
SYSTEM.AP(:, DEF.remaining)    = sum(SYSTEM.Worg.*SYSTEM.AP, 2)./sum(SYSTEM.Worg, ↙
2);
SYSTEM.KpHSA(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.KpHSA, 2)./sum(SYSTEM. ↙
Worg, 2);

%__Subcompartment volume [L]_____________________________________________________
%%% fraction of vascular space of each tissue %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%values are from Gill et al., 2016
%thymus is from Shah and Betts, 2012
%gonads is calculated from the Vint (Valentin, 2002)
```

```
SYSTEM.FraVas = ones(STUDY.IndNo, SYSTEM.CompNo);

SYSTEM.FraVas(:, DEF.lung)      = 0.185*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.adipose)   = 0.031*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.bone)      = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.brain)     = -0.000545 .* Age + 0.056;
SYSTEM.FraVas(:, DEF.gonads)    = 0.069*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.heart)     = 0.042*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.kidney)    = 0.07*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.muscle)    = 0.027*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.skin)      = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.thymus)    = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.gut)       = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.spleen)    = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.pancreas)  = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.liver)     = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.lymphnode) = 0.05*ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.plasma)    = ones(STUDY.IndNo, 1);
SYSTEM.FraVas(:, DEF.RBC)       = ones(STUDY.IndNo, 1);

%weighted mean for the remaining organ
SYSTEM.FraVas(:, DEF.remaining) = sum(SYSTEM.Worg.*SYSTEM.FraVas, 2)./...
                                 sum(SYSTEM.Worg, 2);

SYSTEM.Vvas = SYSTEM.FraVas.*SYSTEM.Vorg;
SYSTEM.Vint = (SYSTEM.Vorg.*SYSTEM.FraEW) - (SYSTEM.Vvas.*(1 - SYSTEM.HCT));
SYSTEM.Vcel = SYSTEM.Vorg - SYSTEM.Vvas - SYSTEM.Vint;

%__pH of the each organ_________________________________________________________
%values are from Schmitt, 2008
SYSTEM.pH = zeros(1, SYSTEM.OrgNo);

SYSTEM.pH(DEF.lung)      = 6.6;
SYSTEM.pH(DEF.adipose)   = 7.1;
SYSTEM.pH(DEF.bone)      = 7.0;
SYSTEM.pH(DEF.brain)     = 7.1;
SYSTEM.pH(DEF.gonads)    = 7.0;
SYSTEM.pH(DEF.heart)     = 7.1;
SYSTEM.pH(DEF.kidney)    = 7.22;
SYSTEM.pH(DEF.muscle)    = 7.0;
SYSTEM.pH(DEF.skin)      = 7.0;
SYSTEM.pH(DEF.thymus)    = 7.0;  %no data; take global value of Rodgers % Rowland, ↙
2005
SYSTEM.pH(DEF.gut)       = 7.0;
SYSTEM.pH(DEF.spleen)    = 7.0;
SYSTEM.pH(DEF.pancreas)  = 7.0;  %no data; take global value of Rodgers % Rowland, ↙
2005
SYSTEM.pH(DEF.liver)     = 7.23;
SYSTEM.pH(DEF.lymphnode) = 7.0;
SYSTEM.pH(DEF.plasma)    = 7.4;  %Valentin, 2002
SYSTEM.pH(DEF.RBC)       = 7.21; %Waddell, 1969
SYSTEM.pH(DEF.remaining) = 7.0;  %no data; take global value of Rodgers % Rowland, ↙
2005
```

```
%__blood flows [L/h]______________________________________________________________
%data are from Stader et al., 2018
FraQorg    = zeros(STUDY.IndNo, SYSTEM.OrgNo);
SYSTEM.Qorg = zeros(STUDY.IndNo, SYSTEM.OrgNo);

%%% Cardiac output (CO) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CO_Mean = 159.*SYSTEM.BSA - 1.56.*Age + 114;

CO = normrnd(CO_Mean, (21.1/100).*CO_Mean);

%%% regional blood flows - fraction of CO %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
FraQorg(:, DEF.lung)       = 100*ones(STUDY.IndNo, 1);

FraQorg(:, DEF.adipose)    = (0.044 + 0.027.*Sex) .* Age + 2.4.*Sex + 3.9;

FraQorg(:, DEF.bone)       = 5*ones(STUDY.IndNo, 1);

FraQorg(:, DEF.brain)      = exp(-0.48.*SYSTEM.BSA + 0.04.*Sex + 3.5);

FraQorg(:, DEF.gonads)     = -0.03.*Sex + 0.05;

FraQorg(:, DEF.heart)      = 4 + 1.*Sex;

FraQorg(:, DEF.kidney)     = -8.7.*SYSTEM.BSA + 0.29.*SYSTEM.Height -...
                              0.081.*Age - 13;

FraQorg(:, DEF.muscle)     = -6.4.*Sex + 17.5;

FraQorg(:, DEF.skin)       = 5*ones(STUDY.IndNo, 1);

FraQorg(:, DEF.thymus)     = 1.5*ones(STUDY.IndNo, 1);

FraQorg(:, DEF.liver)      = -0.108.*Age + 1.04.*Sex + 27.9;

%hepatic arterial blood flow appears to be independent of age
FraQHA                       = 6.5*ones(STUDY.IndNo,1);
FraQPV                       = FraQorg(:, DEF.liver) - FraQHA;

%values of gut, spleen and pancreas are scaled via hepatic blood flow from ICRP
FraQorg(:, DEF.gut)        = ((2.*Sex + 14).*FraQPV)./(1.5.*Sex + 19);
FraQorg(:, DEF.spleen)     = (3.*FraQPV)./(1.5.*Sex + 19);
FraQorg(:, DEF.pancreas)   = (1.*FraQPV)./(1.5.*Sex + 19);

%blood flow bypassing the portal vein organs gut, spleen and pancreas
FraQBY                       = FraQPV  -...
                              FraQorg(:, DEF.gut) -...
                              FraQorg(:, DEF.spleen) -...
                              FraQorg(:, DEF.pancreas);

FraQorg(:, DEF.lymphnode) = 1.65*ones(STUDY.IndNo, 1);

FraQorg(:, DEF.remaining) = FraQorg(:, DEF.lung)    -...
```

```
                             FraQorg(:, DEF.adipose) -...
                             FraQorg(:, DEF.bone)    -...
                             FraQorg(:, DEF.gonads)  -...
                             FraQorg(:, DEF.heart)   -...
                             FraQorg(:, DEF.kidney)  -...
                             FraQorg(:, DEF.muscle)  -...
                             FraQorg(:, DEF.skin)    -...
                             FraQorg(:, DEF.thymus)  -...
                             FraQorg(:, DEF.liver)   -...
                             FraQorg(:, DEF.lymphnode);

%divide the fraction of blood flows by 100
FraQorg = FraQorg./100;

%%% regional blood flows %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SYSTEM.Qorg = FraQorg.*CO;
SYSTEM.QHA  = (FraQHA./100).*CO;
SYSTEM.QBY  = (FraQBY./100).*CO;

%__lymph flows in [L/h]___________________________________________________________
%data are from Gill et al., 2016

%%% total lymph flow %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
LT = 0.00386.*SYSTEM.Weight;

SYSTEM.TotLymphFlow = normrnd(LT, (30/100).*LT);

%%% fraction of regional lymph flows %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
FraLorg = zeros(1, SYSTEM.OrgNo);

FraLorg(DEF.lung)      = 0.03;
FraLorg(DEF.adipose)   = 0.128;
FraLorg(DEF.bone)      = 0;
FraLorg(DEF.brain)     = 0.0105;
FraLorg(DEF.gonads)    = 0.013;
FraLorg(DEF.heart)     = 0.01;
FraLorg(DEF.kidney)    = 0.085;
FraLorg(DEF.muscle)    = 0.16;
FraLorg(DEF.skin)      = 0.073;
FraLorg(DEF.thymus)    = 0.011;
FraLorg(DEF.gut)       = 0.12;
FraLorg(DEF.spleen)    = 0;
FraLorg(DEF.pancreas)  = 0.003;
FraLorg(DEF.liver)     = 0.33;
FraLorg(DEF.lymphnode) = 0;

%it is assumed that the fraction of lymph flow is similar for all individuals
FraLorg = repmat(FraLorg, STUDY.IndNo, 1);

%the remainig organ get the rest of the lymph flow
FraLorg(:, DEF.remaining) = 1 -...
                             FraLorg(:, DEF.lung)     -...
                             FraLorg(:, DEF.adipose)  -...
```

```matlab
                              FraLorg(:, DEF.bone)      -...
                              FraLorg(:, DEF.brain)     -...
                              FraLorg(:, DEF.gonads)    -...
                              FraLorg(:, DEF.heart)     -...
                              FraLorg(:, DEF.kidney)    -...
                              FraLorg(:, DEF.muscle)    -...
                              FraLorg(:, DEF.skin)      -...
                              FraLorg(:, DEF.thymus)    -...
                              FraLorg(:, DEF.gut)       -...
                              FraLorg(:, DEF.spleen)    -...
                              FraLorg(:, DEF.pancreas) -...
                              FraLorg(:, DEF.liver);

%calculate the lymph flow
SYSTEM.Lorg = SYSTEM.TotLymphFlow.*FraLorg;

%__liver parameters______________________________________________________________
%%% MPPGL %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%MPPGL = microsomal protein per gram liver in [mg / g liver]
%Barter et al., 2008

MPPGL_Mean = 10.^(0.0000024.*Age.^3 - 0.00038.*Age.^2 + 0.0158.*Age + 1.407);
MPPGL_CV   = 46;
MPPGL_Min  = [10, 10];
MPPGL_Max  = [110, 110];

SYSTEM.MPPGL = Calc_SysPar(MPPGL_Mean, MPPGL_CV, MPPGL_Min, MPPGL_Max);

%__hepatic CYP enzyme abundance [pmol/mg]_______________________________________
%hepatic CYP abundance is from Achour et al., 2014
CYP2D6_Mean = 12.6*ones(STUDY.IndNo, 1);
CYP2D6_CV   = 74;
CYP2D6_Min  = [4.2, 4.2];
CYP2D6_Max  = [38, 38];

CYP2J2_Mean = 1.2*ones(STUDY.IndNo, 1);
CYP2J2_CV   = 58;
CYP2J2_Min  = [0.4, 0.4];
CYP2J2_Max  = [3.6, 3.6];

CYP3A4_Mean = 93.0*ones(STUDY.IndNo, 1);
CYP3A4_CV   = 81;
CYP3A4_Min  = [18.6, 18.6];
CYP3A4_Max  = [601 601];

%AhC = Abundance of hepatic CYP enzymes
AhC(:, DEF.CYP2D6)  = Calc_SysPar(CYP2D6_Mean, CYP2D6_CV, CYP2D6_Min, CYP2D6_Max);
AhC(:, DEF.CYP3A4)  = Calc_SysPar(CYP3A4_Mean, CYP3A4_CV, CYP3A4_Min, CYP3A4_Max);
AhC(:, DEF.CYP3A5)  = 0.41.*AhC(:, DEF.CYP3A4) + 56.1;
AhC(:, DEF.CYP2J2)  = Calc_SysPar(CYP2J2_Mean, CYP2J2_CV, CYP2J2_Min, CYP2J2_Max);

SYSTEM.CYPhe_AB = AhC.*PheCYP;
```

```
%degradation rate of hepatic CYP enzymes in [1/h]
SYSTEM.CYPhe_kdeg = zeros(1, SYSTEM.CYPliNo);

SYSTEM.CYPhe_kdeg(DEF.CYP2D6) = 0.0143;
SYSTEM.CYPhe_kdeg(DEF.CYP3A4) = 0.0077;
SYSTEM.CYPhe_kdeg(DEF.CYP3A5) = 0.0193;
SYSTEM.CYPhe_kdeg(DEF.CYP2J2) = 0.01;

SYSTEM.CYPhe_kdeg = repmat(SYSTEM.CYPhe_kdeg, STUDY.IndNo, 1);

%__kidney parameters_________________________________________________________________
%%% glomerular filtration rate (GFR) in [mL/min] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
GFR = exp(-0.0079.*Age + 0.5.*SYSTEM.BSA + 4.2);

SYSTEM.GFR = normrnd(GFR, (14.7/100).*GFR);

%__parameters of the GI tract________________________________________________________
%%% Volumes of intestinal segments %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%total volume of intestinal segments in [L]
%division of the intestinal volume to different segments is given in the ICRP
SYSTEM.VsegCAT = zeros(STUDY.IndNo, SYSTEM.SegNo);

SYSTEM.VsegCAT(:, DEF.stomach)  = 0.153 .* SYSTEM.Vorg(:, DEF.gut);
SYSTEM.VsegCAT(:, DEF.duodenum) = 0.060 .* SYSTEM.Vorg(:, DEF.gut);
SYSTEM.VsegCAT(:, DEF.jejunum)  = 0.279 .* SYSTEM.Vorg(:, DEF.gut);
SYSTEM.VsegCAT(:, DEF.ileum)    = 0.316 .* SYSTEM.Vorg(:, DEF.gut);
SYSTEM.VsegCAT(:, DEF.colon)    = 0.192 .* SYSTEM.Vorg(:, DEF.gut);

%vascular space of the gut in [L]
%Gill et al. (2016) report a fraction of 0.05 for the gut
SYSTEM.VvasCAT = zeros(STUDY.IndNo, SYSTEM.SegNo);

SYSTEM.VvasCAT(:, DEF.duodenum) = 0.05 .* SYSTEM.VsegCAT(:, DEF.duodenum);
SYSTEM.VvasCAT(:, DEF.jejunum)  = 0.05 .* SYSTEM.VsegCAT(:, DEF.jejunum);
SYSTEM.VvasCAT(:, DEF.ileum)    = 0.05 .* SYSTEM.VsegCAT(:, DEF.ileum);
SYSTEM.VvasCAT(:, DEF.colon)    = 0.05 .* SYSTEM.VsegCAT(:, DEF.colon);

SYSTEM.Vvas(:, DEF.gut) = sum(SYSTEM.VvasCAT, 2);

%interstitial space of the gut in [L]
SYSTEM.VintCAT = SYSTEM.FraEW(:, DEF.gut) .* SYSTEM.VsegCAT - SYSTEM.VvasCAT;

%the stomach is not modelled with interstitial space
SYSTEM.VintCAT(:, DEF.stomach) = 0;

SYSTEM.Vint(:, DEF.gut) = sum(SYSTEM.VintCAT, 2);

%%% Length of the intestine in [cm] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%length are from the ICRP (Valentin, 2002)
SYSTEM.LengthGU = zeros(STUDY.IndNo, SYSTEM.SegNo);

SYSTEM.LengthGU(:, DEF.duodenum) = 0.091 .* (1.6.*SYSTEM.Height);
SYSTEM.LengthGU(:, DEF.jejunum)  = 0.426 .* (1.6.*SYSTEM.Height);
SYSTEM.LengthGU(:, DEF.ileum)     = 0.483 .* (1.6.*SYSTEM.Height);
SYSTEM.LengthGU(:, DEF.colon)     = 0.52 .* SYSTEM.Height + 18.5;

%%% Radius of the intestine in [cm] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%it is assumed that the intestine is a cylinder
%1000 converts L to cm³
SYSTEM.RadiusGU = sqrt((SYSTEM.VsegCAT .* 1000) ./ (pi .* SYSTEM.LengthGU));

%%% Surface of the intestine in [cm²] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SurfaceGU = 2 .* pi .* SYSTEM.RadiusGU .* SYSTEM.LengthGU;

%the intestinal surface is enlarged by plicae circulares, villi and microvilli
%data from Helander & Fändriks, 2014
FPC = zeros(1, SYSTEM.SegNo);     %factor for plicae circulares
FVI = zeros(1, SYSTEM.SegNo);     %factor for villi
FMV = zeros(1, SYSTEM.SegNo);     %factor for microvilli

FPC(DEF.duodenum) = 1.6;  FVI(DEF.duodenum) = 6.5;  FMV(DEF.duodenum) = 14.6;
FPC(DEF.jejunum)  = 1.6;  FVI(DEF.jejunum)  = 8.6;  FMV(DEF.jejunum)  =  9.2;
FPC(DEF.ileum)    = 1.6;  FVI(DEF.ileum)    = 4.5;  FMV(DEF.ileum)    = 15.7;
FPC(DEF.colon)    = 1.0;  FVI(DEF.colon)    = 6.5;  FMV(DEF.colon)    = 1.0;

FPC = repmat(FPC, STUDY.IndNo, 1);
FVI = repmat(FVI, STUDY.IndNo, 1);
FMV = repmat(FMV, STUDY.IndNo, 1);

SYSTEM.PSA = SurfaceGU.*FPC.*FVI.*FMV;

%%% enterocyte space of the gut in [L] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SYSTEM.VentCAT = (3040 .* (SurfaceGU ./ 0.0001)) .* 770 .* 1E-15;

%stomach does not have enterocytes
SYSTEM.VentCAT(:, DEF.stomach) = 0;

%luminal space of the gut in [L]
SYSTEM.VlumCAT = SYSTEM.VsegCAT - SYSTEM.VvasCAT - SYSTEM.VintCAT -...
                 SYSTEM.VentCAT;

%%% intestinal transit times [h] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%uniform distribution between 0.25 (Yu & Amidon 1999) and 0.4 (Jamei et al. 2009)
%for the gastric emptying time (GET) - for liquids only
GET = 0.25 + (0.40 - 0.25)*rand(STUDY.IndNo, 1);

SIT_Mean = 3.4*ones(STUDY.IndNo, 1);
SIT_CV   = 40.2;
SIT_Min  = [0.5, 0.5];
SIT_Max  = [9.5, 9.5];
SIT      = Calc_SysPar(SIT_Mean, SIT_CV, SIT_Min, SIT_Max);

CNT_Mean = 17.2*ones(STUDY.IndNo, 1);
CNT_CV   = 20.0;
CNT_Min  = [17.2/5, 17.2/5];
CNT_Max  = [17.2*5, 17.2*5];
```

```
CNT      = Calc_SysPar(CNT_Mean, CNT_CV, CNT_Min, CNT_Max);

%separation into different segments is based on length (Darwich et al., 2010)
SYSTEM.TransitT = zeros(STUDY.IndNo, SYSTEM.SegNo);

SYSTEM.TransitT(:, DEF.stomach)  = GET;
SYSTEM.TransitT(:, DEF.duodenum) = SIT.*0.091;
SYSTEM.TransitT(:, DEF.jejunum)  = SIT.*0.426;
SYSTEM.TransitT(:, DEF.ileum)    = SIT.*0.483;
SYSTEM.TransitT(:, DEF.colon)    = CNT;

%__intestinal CYP enzymes_______________________________________________________
%minimum / maximum is an arbitrary 3-fold difference
CYP2D6_Mean = 0.8*ones(STUDY.IndNo, 1);
CYP2D6_CV   = 60;
CYP2D6_Min  = [0.8/3, 0.8/3];
CYP2D6_Max  = [0.8*3, 0.8*3];

CYP3A4_Mean = 66.2*ones(STUDY.IndNo, 1);
CYP3A4_CV   = 60;
CYP3A4_Min  = [66.3/3, 66.3/3];
CYP3A4_Max  = [66.3*3, 66.3*3];

CYP3A5_Mean = 24.6*ones(STUDY.IndNo, 1);
CYP3A5_CV   = 60;
CYP3A5_Min  = [24.6/3, 24.6/3];
CYP3A5_Max  = [24.6*3, 24.6*3];

%AiC = Abundance of intestinal CYP enzymes
AiC(:, DEF.CYP2D6)  = Calc_SysPar(CYP2D6_Mean, CYP2D6_CV, CYP2D6_Min, CYP2D6_Max);
AiC(:, DEF.CYP3A4)  = Calc_SysPar(CYP3A4_Mean, CYP3A4_CV, CYP3A4_Min, CYP3A4_Max);
AiC(:, DEF.CYP3A5)  = Calc_SysPar(CYP3A5_Mean, CYP3A5_CV, CYP3A5_Min, CYP3A5_Max);

%intestinal CYP abundance [nmol]
CYPin_AB = AiC .* PheCYP(:, 1:SYSTEM.CYPinNo);

%separation into different segments is done according to Paine et al., 1997
SYSTEM.CYPseg_AB = zeros(STUDY.IndNo, SYSTEM.CYPinNo, SYSTEM.SegNo);
SYSTEM.CYPseg_AB(:, :, DEF.duodenum) = 0.136.*CYPin_AB;
SYSTEM.CYPseg_AB(:, :, DEF.jejunum)  = 0.544.*CYPin_AB;
SYSTEM.CYPseg_AB(:, :, DEF.ileum)    = 0.320.*CYPin_AB;

%degradation rate of intestinal CYP enzymes in [1/h]
SYSTEM.CYPin_kdeg = 0.03.*ones(STUDY.IndNo, SYSTEM.CYPliNo);

%===============================================================================
%%% USED FUNCTION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%===============================================================================
function SysPar = Calc_SysPar(Mean, CV, Min, Max)

    %read gender differences
    MinMale = Min(1);    MinFemale = Min(2);
    MaxMale = Max(1);    MaxFemale = Max(2);
    %generate parameter
    SysPar = normrnd(Mean, (CV/100).*Mean);

    %truncate parameter at minimum and maximum observed values
    for sub = 1:STUDY.IndNo
        if SYSTEM.Sex(sub) == 1
            if SysPar(sub) < MinFemale || SysPar(sub) > MaxFemale
                SysPar(sub) = normrnd(Mean(sub), (CV/1000).*Mean(sub));
            end

        else
            if SysPar(sub) < MinMale || SysPar(sub) > MaxMale
                SysPar(sub) = normrnd(Mean(sub), (CV/1000).*Mean(sub));
            end
        end
    end
end

end

%===============================================================================
%%% USED REFERENCES %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%===============================================================================
% Stader F, Siccardi M, Battegay M, Kinvig H, Penny MA, & Marzolini C. Repository ↙
describing an aging population to inform physiologically based pharmacokinetic ↙
models considering anatomical, physiological, and biological age-dependent changes. ↙
Clinical Pharmacokinetics, 2018. [Epub ahead of print].
% Gill KL, Gardner I, Li L, & Jamei M. A bottom-up whole-body physiologically based ↙
pharmacokinetic model to mechanistically predict tissue distribution and the rate ↙
of subcutaneous absorption of therapeutic proteins. The AAPS Journal, 2016. 18(1): ↙
156-170.
% Valentin J. Basic anatomical and physiological data for use in radiological ↙
protection: reference values: ICRP Publication 89. Annals of the ICRP, 2002. 32(3): ↙
1-277.
% Snyder W, Cook M, Nasset E, Karhausen L, Howells GP, & Tipton I, Report of the ↙
Task Group on Reference Man. 1975, Oxford (UK): Pergamon Press.
% Heinemann A, Wischhusen F, Püschel K, & Rogiers X. Standard liver volume in the ↙
Caucasian population. Liver Transplantation and Surgery, 1999. 5(5): 366-368.
% Jamei M, Dickinson GL, & Rostami-Hodjegan A. A framework for assessing inter- ↙
individual variability in pharmacokinetics using virtual human populations and ↙
integrating general knowledge of physical chemistry, biology, anatomy, physiology ↙
and genetics: a tale of ‘bottom-up’vs ‘top-down’recognition of covariates. Drug ↙
Metabolism and Pharmacokinetics, 2009. 24(1): 53-75.
% Rodgers T, Leahy D, & Rowland M. Tissue distribution of basic drugs: Accounting ↙
for enantiomeric, compound and regional differences amongst ??blocking drugs in ↙
rat. Journal of Pharmaceutical Sciences, 2005. 94(6): 1237-1248.
% Pierson R, Price DC, Wang J, & Jain RK. Extracellular water measurements: organ ↙
tracer kinetics of bromide and sucrose in rats and man. American Journal of ↙
Physiology-Renal Physiology, 1978. 235(3): F254-F264.
% Bieri J & Prival E. Lipid composition of testes from various species. Comparative ↙
Biochemistry and Physiology, 1965. 15(3): 275-282.
% Diagne A, Fauvel J, Record M, Chap H, & Douste-Blazy L. Studies on ether ↙
```

```
phospholipids: II. Comparative composition of various tissues from human, rat and ↙
guinea pig. Biochimica et Biophysica Acta, 1984. 793(2): 221-231.
% Zhu H, Melder RJ, Baxter LT, & Jain RK. Physiologically based kinetic model of ↙
effector cell biodistribution in mammals: implications for adoptive immunotherapy. ↙
Cancer Research, 1996. 56(16): 3771-3781.
% Yan SH & Franks J. Albumin metabolism in elderly men and women. The Journal of ↙
Laboratory and Clinical Medicine, 1968. 72(3): 449-454.
% Shah DK & Betts AM. Towards a platform PBPK model to characterize the plasma and ↙
tissue disposition of monoclonal antibodies in preclinical species and human. ↙
Journal of Pharmacokinetics and Pharmacodynamics, 2012. 39(1): 67-86.
% Schmitt W. General approach for the calculation of tissue to plasma partition ↙
coefficients. Toxicology in Vitro, 2008. 22(2): 457-467.
% Waddell WJ & Bates RG. Intracellular pH. Physiological Reviews, 1969. 49(2): 285-↙
329.
% Barter ZE, Chowdry JE, Harlow JR, Snawder JE, Lipscomb JC, & Rostami-Hodjegan A. ↙
Covariation of human microsomal protein per gram of liver with age: absence of ↙
influence of operator and sample storage may justify interlaboratory data pooling. ↙
Drug Metabolism and Disposition, 2008. 36(12): 2405-2409.
% Achour B, Barber J, & Rostami-Hodjegan A. Expression of hepatic drug-metabolizing↙
cytochrome p450 enzymes and their intercorrelations: a meta-analysis. Drug ↙
Metabolism and Disposition, 2014. 42(8): 1349-1356.
% Helander HF & Fändriks L. Surface area of the digestive tract-revisited. ↙
Scandinavian Journal of Gastroenterology, 2014. 49(6): 681-689.
% Yu LX & Amidon GL. A compartmental absorption and transit model for estimating ↙
oral drug absorption. International Journal of Pharmaceutics, 1999. 186(2): 119- ↙
125.
% Darwich A, Neuhoff S, Jamei M, & Rostami-Hodjegan A. Interplay of metabolism and ↙
transport in determining oral drug absorption and gut wall metabolism: a simulation ↙
assessment using the “Advanced Dissolution, Absorption, Metabolism (ADAM)” model. ↙
Current Drug Metabolism, 2010. 11(9): 716-729.
% Jamei M, Turner D, Yang J, Neuhoff S, Polak S, Rostami-Hodjegan A, & Tucker G. ↙
Population-based mechanistic prediction of oral drug absorption. The AAPS Journal, ↙
2009. 11(2): 225-237.
% Paine MF, Khalighi M, Fisher JM, Shen DD, Kunze KL, Marsh CL, Perkins JD, & ↙
Thummel KE. Characterization of interintestinal and intraintestinal variations in ↙
human CYP3A-dependent metabolism. Journal of Pharmacology and Experimental ↙
Therapeutics, 1997. 283(3): 1552-1562.
```

```
function[] = Drug()
%This function loads the revelant drug files from the drug file library and
%performs the in vitro-to-in vivo extrapolation

global DEF       %global DEF defines model parameters
global SYSTEM    %global SYSTEM defines sytem parameters
global DRUG      %global DRUG defines drug parameters
global DDI       %global DDI enhances drug parameters for DDI prediction
global STUDY     %global STUDY defines study design parameters
global MODEL     %global MODEL defines parameters important for modeling

%define variables used in the script
dru = 5;

%__PreProcessing_________________________________________________________________
%prepare all drug parameters for each drug file in the library

%%% physchem characteristics %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DRUG.MolW           = zeros(1, DEF.DrugLibNo);    %molecular weight in [g/mol]
DRUG.logP           = zeros(1, DEF.DrugLibNo);    %octanol water partition coef
DRUG.pka            = zeros(1, DEF.DrugLibNo);
DRUG.BP             = zeros(1, DEF.DrugLibNo);    %blood-to-plasma ratio
DRUG.fu             = zeros(1, DEF.DrugLibNo);    %fraction unbound
DRUG.protein        = zeros(1, DEF.DrugLibNo);    %main binding protein

%%% absorption %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%apparent permeability in 10^-6 cm/sec
DRUG.Papp           = zeros(1, DEF.DrugLibNo);

%a rate constant can be introduced to match the observed Tmax
DRUG.LagRate        = zeros(1, DEF.DrugLibNo);

%%% distribution %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%universal KP Scalar for all compartments at once
DRUG.KpScalarAll    = zeros(1, DEF.DrugLibNo);

%Kp scalar for single compartments
DRUG.KpScalar       = zeros(SYSTEM.CompNo, DEF.DrugLibNo);

%%% metabolism & elimination %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DRUG.Vmax_CYP       = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);
DRUG.Km_CYP         = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);
DRUG.CLint_CYP      = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);

DRUG.CLint          = zeros(1, DEF.DrugLibNo);

DRUG.CLrenal        = zeros(1, DEF.DrugLibNo);
DRUG.CLrenalCV      = zeros(1, DEF.DrugLibNo);

DRUG.CLbile         = zeros(1, DEF.DrugLibNo);
DRUG.CLbileCV       = zeros(1, DEF.DrugLibNo);

DRUG.CLadditional   = zeros(1, DEF.DrugLibNo);
DRUG.CLadditionalCV = zeros(1, DEF.DrugLibNo);

%%% drug-drug interactions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DRUG.Ki_CYP         = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);

DRUG.kinact_CYP     = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);
DRUG.Kapp_CYP       = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);

DRUG.IndMax_CYP     = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);
DRUG.IC50_CYP       = zeros(SYSTEM.CYPliNo, DEF.DrugLibNo);

%__load the drug models__________________________________________________________
%load the user-defined drug model
addpath('DrugLibrary');

%convert the string of the user chosen drug model to the function in the library
for d = 1:DRUG.DrugNo
    switch d
        case 1
            drug1 = str2func(DEF.name{dru, DRUG.DrugName(d)});
            drug1();

        case 2
            drug2 = str2func(DEF.name{dru, DRUG.DrugName(d)});
            drug2();

        case 3
            drug3 = str2func(DEF.name{dru, DRUG.DrugName(d)});
            drug3();
    end
end

%__PostProcessing________________________________________________________________
%Extract only non-zero values for used drugs in the simulation based on MW
ParGen    = [DRUG.MolW; DRUG.logP; DRUG.pka; DRUG.BP; DRUG.fu; DRUG.protein; ...
             DRUG.Papp; DRUG.LagRate; ...
             DRUG.KpScalarAll; ...
             DRUG.CLint; DRUG.CLrenal; DRUG.CLrenalCV;...
             DRUG.CLbile; DRUG.CLbileCV; ...
             DRUG.CLadditional; DRUG.CLadditionalCV];
ParSystem = [DRUG.KpScalar; DRUG.MolW];
ParCYP    = [DRUG.Vmax_CYP; DRUG.Km_CYP; DRUG.CLint_CYP;...
             DRUG.Ki_CYP; DRUG.kinact_CYP; DRUG.Kapp_CYP;...
             DRUG.IndMax_CYP; DRUG.IC50_CYP; DRUG.MolW; ];

%delete zero values
ParGen(:, all (~any (ParGen), 1))       = [];
ParSystem(:, all (~any (ParSystem), 1)) = [];
ParCYP(:, all (~any (ParCYP), 1))       = [];

%factor for DDI studies
if DRUG.DrugNo == 1
    %in the case of only one drug, only auto-induction and inhibition is used
    DDI.DDINo = 1;

else
    DDI.DDINo = 2*DRUG.DrugNo;
end

FacDDI = DDI.DDINo / DRUG.DrugNo;

DDI.MolW         = repmat(ParGen(1, :), 1, FacDDI);
DDI.logP         = repmat(ParGen(2, :), 1, FacDDI);
DDI.pka          = repmat(ParGen(3, :), 1, FacDDI);
DDI.BP           = repmat(ParGen(4, :), 1, FacDDI);
DDI.fu           = repmat(ParGen(5, :), 1, FacDDI);
DDI.protein      = repmat(ParGen(6, :), 1, FacDDI);

DDI.Papp         = repmat(ParGen(7, :), 1, FacDDI);
DDI.LagRate      = repmat(ParGen(8, :), 1, FacDDI);
%if there is no delay, the lag rate is set to a very high value
DDI.LagRate(DDI.LagRate == 0) = 1000;

DDI.KpScalarAll = repmat(ParGen(9, :), 1, FacDDI);
DDI.KpScalar     = repmat(ParSystem(1:SYSTEM.CompNo, :), 1, FacDDI);

%Kp Scalar cannot be 0, because they are multiplied to Kpu
DDI.KpScalarAll(DDI.KpScalarAll == 0) = 1;
DDI.KpScalar(DDI.KpScalar == 0) = 1;

%Vmax is converted from pmol/min/pmol to micromol/h/pmol
DDI.Vmax_CYP     = repmat(ParCYP(1:SYSTEM.CYPliNo, :), 1, FacDDI) .*10^-6 .* 60;

%if Km is 0, it is converted to 1 to prevent dividing by 0 in the code
DDI.Km_CYP       = repmat(ParCYP(SYSTEM.CYPliNo+1:2*SYSTEM.CYPliNo, :), 1, FacDDI);
DDI.Km_CYP(DDI.Km_CYP == 0) = 1;

%CLint (CYP) is converted from microL/min/pmol to L/h/pmol
DDI.CLint_CYP    = repmat(ParCYP(2*SYSTEM.CYPliNo+1:3*SYSTEM.CYPliNo, :),...
                         1, FacDDI) .*10^-6 .* 60;

%CLint (tot hep) is converted from microL/min/mg to L/h/mg
DDI.CLint        = repmat(ParGen(10, :), 1, FacDDI) .*10^-6 .* 60;

DDI.CLrenal      = repmat(ParGen(11, :), 1, FacDDI);
DDI.CLrenalCV    = repmat(ParGen(12, :), 1, FacDDI);
DDI.CLbile       = repmat(ParGen(13, :), 1, FacDDI);
DDI.CLbileCV     = repmat(ParGen(14, :), 1, FacDDI);
DDI.CLadd        = repmat(ParGen(15, :), 1, FacDDI);
DDI.CLaddCV      = repmat(ParGen(16, :), 1, FacDDI);

DDI.Ki_CYP       = repmat(ParCYP(3*SYSTEM.CYPliNo+1:4*SYSTEM.CYPliNo, :), 1,↙
FacDDI);
DDI.kinact_CYP   = repmat(ParCYP(4*SYSTEM.CYPliNo+1:5*SYSTEM.CYPliNo, :), 1,↙
FacDDI);
DDI.Kapp_CYP     = repmat(ParCYP(5*SYSTEM.CYPliNo+1:6*SYSTEM.CYPliNo, :), 1,↙
FacDDI);
DDI.IndMax_CYP  = repmat(ParCYP(6*SYSTEM.CYPliNo+1:7*SYSTEM.CYPliNo, :), 1, ↙
FacDDI);
DDI.IC50_CYP    = repmat(ParCYP(7*SYSTEM.CYPliNo+1:8*SYSTEM.CYPliNo, :), 1, ↙
FacDDI);

%if Ki / Kapp / IC50 is 0, it is converted to 1 to prevent dividing by 0 in the ↙
code
DDI.Ki_CYP(DDI.Ki_CYP == 0)  = 1;
DDI.Kapp_CYP(DDI.Kapp_CYP == 0)  = 1;
DDI.IC50_CYP(DDI.IC50_CYP == 0)  = 1;

%%% name of drugs for outputs %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DrugName = cell(DRUG.DrugNo, 1);    DDIName = cell(DDI.DDINo, 1);

%save the name of the drug into a new vector
for d = 1:DRUG.DrugNo
    DrugName{d} = DEF.name{5, DRUG.DrugName(d)};
end

%DDI predictions need a combined name
Help = ' + ';

for d = 1:DRUG.DrugNo
    switch d
        case 1
            DDIName{d}   = DrugName{d};
        case 2
            DDIName{d}   = DrugName{d};
            DDIName{d+1} = [DrugName{d-1}, Help, DrugName{d}];
            DDIName{d+2} = [DrugName{d}, Help, DrugName{d-1}];
        case 3
            DDIName{d}   = DrugName{d};
            DDIName{d+1} = [DrugName{d-2}, Help, DrugName{d-1} , Help,DrugName{d}];
            DDIName{d+2} = [DrugName{d-1}, Help, DrugName{d-2} , Help,DrugName{d}];
            DDIName{d+3} = [DrugName{d}, Help, DrugName{d-2} ,Help, DrugName{d-1}];
    end
end

%save the name for DDI predictions globally
DDI.Name = DDIName;

%__drug absorption______________________________________________________________
%%% Effective permeability %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%in vitro - in vivo relationship comes from Sun et al., 2002
PeffMen = 10.^ (0.6795 .* log10(DDI.Papp) - 0.3355);
PeffMen = permute( repmat(PeffMen, STUDY.IndNo, 1, SYSTEM.SegNo), [1, 3, 2]);

%PeffAll for each of the segments will be based on the difference in length
%following the approach by Darwich et al., 2010
PeffMen = PeffMen .* (SYSTEM.LengthGU ./ sum(SYSTEM.LengthGU, 2));

%absorption clearance, which will be used in the model
```

```
DDI.CLab = repmat(SYSTEM.PSA, 1, 1, DDI.DDINo) .* PeffMen .* 3600 .* 0.001;


%__drug distribution_____________________________________________________________
%%% Define plasma concentration of the main binding protein

%concentration of the plasma-binding protein in [g/L]
ProtConc = zeros(STUDY.IndNo, DDI.DDINo);

%reference for the calculation of age-dependency (reference is 30 years)
ProtRef  = zeros(STUDY.IndNo, DDI.DDINo);

%partition coefficient for plasma-binding proteins into the tissue
KpPR = zeros(STUDY.IndNo, SYSTEM.CompNo, DDI.DDINo);

for d = 1:DDI.DDINo
    if DDI.protein(d) == DEF.albumin
        ProtConc(:, d)  = SYSTEM.Albumin;
        ProtRef(:, d)   = 45.6;
        KpPR(:, :, d)   = SYSTEM.KpHSA;

    elseif DDI.protein(d) == DEF.AAG
        ProtConc(:, d)  = SYSTEM.AAG;
        ProtRef(:, d)   = 0.798;
        KpPR(:, :, d)   = SYSTEM.KpHSA;
    end
end

%%% pH-dependent parameters for distribution (Rodgers & Rowland) %%%%%%%%%%%%%%
%this are the X,Y and Z parameter from Rodgers & Rowland
DDI.KRio = zeros(SYSTEM.CompNo, DDI.DDINo);

for com = 1:SYSTEM.CompNo
    for d = 1:DDI.DDINo
        DDI.KRio(com, d) = 1 + 10.^(DDI.pka(d) - SYSTEM.pH(com));
    end
end

%%% vegetable oil:water partition coefficient %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% prediction according to Poulin & Theil (2002), because it is assumed that
%it predicts partitioning into the adipose tissue better
DDI.logD = 1.115.*(abs(DDI.logP)) - 1.35 - log10(DDI.KRio(DEF.plasma, :));

%%% fraction unbound in plasma - age-dependent changes %%%%%%%%%%%%%%%%%%%%%%%%%
DDI.fup = 1 ./ (1 + ((((1./(repmat(DDI.fu, STUDY.IndNo, 1))) - 1)./...
          ProtRef).*ProtConc));

%%% fraction unbound in the interstitial space %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%fuint is based on the semiphysiological Bmax model
%it is only calculated for albumin, because AAG doesn't distribute to Vint

DDI.fuine = ones(STUDY.IndNo, SYSTEM.CompNo, DDI.DDINo);
for d = 1:DDI.DDINo
```

```
    if DDI.protein(d) == DEF.albumin
        DDI.fuine(:, :, d) = 1 ./ (((SYSTEM.KpHSA./SYSTEM.FraEW) .*...
                                    ((1./DDI.fup(:,d)) - 1)) + 1);
    end
end

%%% fraction unbound in the cellular space %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%fucel is based on Rodgers & Rowland
%it is only calculated for albumin, because AAG does not need to be considered

DDI.fucel = ones(STUDY.IndNo, SYSTEM.CompNo, DDI.DDINo);
for d = 1:DDI.DDINo
    if DDI.protein(d) == DEF.albumin
        DDI.fucel(:, :, d) = 1 ./ (1 + (((DDI.logP(d).*SYSTEM.FraNL +...
                             (0.3.*DDI.logP(d) + 0.7).*SYSTEM.FraNP)./...
                             DDI.KRio(DEF.plasma, d)) +...
                             SYSTEM.KpHSA.*ProtConc(:, d)));
    end
end

%%% tissue-partition coefficient %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DDI.KaPR    = zeros(STUDY.IndNo, DDI.DDINo);
DDI.Kpu = zeros(STUDY.IndNo, SYSTEM.CompNo, DDI.DDINo);

for d = 1:DDI.DDINo

    for ind = 1:STUDY.IndNo
        DDI.KaPR(ind, d) = ((1 ./ DDI.fup(ind, d)) - 1 -...
                            ((DDI.logP(d) .* SYSTEM.FraNL(ind, DEF.plasma) + ((0.3.↙
*DDI.logP(d) + 0.7) .* SYSTEM.FraNP(ind, DEF.plasma))) ./ DDI.KRio(DEF.plasma, d)))↙
.*...
                           (1 ./ ProtConc(ind, d));

        for com = 1:SYSTEM.CompNo
            DDI.Kpu(ind, com, d) = (((DDI.KRio(com, d) .* SYSTEM.FraIW(ind, com)) .↙
/ DDI.KRio(DEF.plasma, d)) +...
                                    SYSTEM.FraEW(ind, com) +...
                                    ((DDI.logP(d) .* SYSTEM.FraNL(ind, com) + (0.3↙
.* DDI.logP(d) + 0.7) .* SYSTEM.FraNP(ind, com)) ./ DDI.KRio(DEF.plasma, d)) +...
                                    (DDI.KaPR(ind, d) .* KpPR(ind, com, d) *↙
ProtConc(ind, d))) .*...
                                    DDI.KpScalarAll(d) .* DDI.KpScalar(com, d);
        end

        DRUG.Kpu(ind, DEF.adipose, d) = (((DDI.KRio(DEF.adipose, d) .* SYSTEM.FraIW↙
(ind, DEF.adipose)) ./ DDI.KRio(DEF.plasma, d)) +...
                                         SYSTEM.FraEW(ind, DEF.adipose) +...
                                         ((DDI.logD(d) .* SYSTEM.FraNL(ind, DEF.↙
adipose) + (0.3 .* DDI.logD(d) + 0.7) .* SYSTEM.FraNP(ind, DEF.adipose)) ./ DDI.↙
KRio(DEF.plasma, d)) +...
                                         (DDI.KaPR(ind, d) .* KpPR(ind, DEF.↙
adipose, d) * ProtConc(ind, d))) .*...
                                         DDI.KpScalarAll(d) .* DDI.KpScalar(DEF.↙
adipose, d);
    end
end

%%% flux through the membrane %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DDI.Jout = repmat(SYSTEM.Qorg, 1, 1, DDI.DDINo);

%prepare variable for faster calculations
BP = permute (repmat (DDI.BP, STUDY.IndNo, 1, SYSTEM.OrgNo), [1, 3, 2]);

DDI.Jin  = (abs((DDI.Kpu(:, 1:16, :) - (1./BP)) .* (DDI.fucel(:, 1:16, :) ./...
            DDI.fuine(:, 1:16, :))) .* DDI.Jout);

%%% Volume of distribution [L/kg] %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
DDI.Vss = zeros(STUDY.IndNo, DDI.DDINo);

for d = 1:DDI.DDINo
    DDI.Vss(:, d) = ((SYSTEM.Vorg(:, DEF.plasma) ./ DDI.fup(:, d)) +...
                      (sum(SYSTEM.Vorg(:, 1:SYSTEM.OrgNo) .*...
                      DDI.Kpu(:, 1:SYSTEM.OrgNo, d), 2)))./SYSTEM.Weight;
end

%__drug elimination______________________________________________________________
%prepare empyt vectors
DDI.CLre = zeros(STUDY.IndNo, DDI.DDINo);
DDI.CLbi = zeros(STUDY.IndNo, DDI.DDINo);
DDI.CLad = zeros(STUDY.IndNo, DDI.DDINo);

for d = 1:DDI.DDINo
    %renal clearance in [L/h]
    DDI.CLre(:, d) = normrnd(DDI.CLrenal(d),...
                             ((DDI.CLrenalCV(d)/100) .* DDI.CLrenal(d)),...
                             STUDY.IndNo, 1);

    %biliary clearance in [L/h]
    DDI.CLbi(:, d) = normrnd(DDI.CLbile(d),...
                             ((DDI.CLbileCV(d)./100) .* DDI.CLbile(d)),...
                             STUDY.IndNo, 1);

    %additional plasma clearance in [L/h]
    DDI.CLad(:, d) = normrnd(DDI.CLadd (d),...
                             ((DDI.CLaddCV(d)./100) .* DDI.CLadd (d)),...
                             STUDY.IndNo, 1);
end

%link the renal clearance to the GFR of each individual
for d = DDI.DDINo
    DDI.CLre(:, d) = DDI.CLre(:, d) .*...
                          (SYSTEM.GFR ./ (130 - 10.*SYSTEM.Sex)) .*...
                          (DDI.fup(:, d)./DDI.fu(d));
end

%ritonavir has an impact on the renal clearance of rivaroxaban, which is not
```

```
%mechanistically in the model, but will be considered
global Index
for dr = 1:DRUG.DrugNo
    if DRUG.DrugName(dr) == DEF.rivaroxaban
        if DRUG.DrugName(dr-1) == DEF.ritonavir
            Index = find(DRUG.DrugName == DEF.rivaroxaban);
            DDI.CLre(:, Index + DRUG.DrugNo) = 0.5 .* DDI.CLre(:, Index + DRUG. ↙
DrugNo);
        end
    end
end

%__prepare DDI matrix________________________________________________________________
%%% Variables for DDI matrix %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%define three different drugs A-c
drugA = 1;    drugB = 2;    drugC = 3;

%six different DDIs are possible
ddi_1 = 1;    %drug A alone
ddi_2 = 2;    %drug B alone
ddi_3 = 3;    %drug A with drug B / drug C alone
ddi_4 = 4;    %drug B with drug A / drug A with drug B and C
ddi_5 = 5;    %drug B with drug A and C
ddi_6 = 6;    %drug C with drug A and B

%abbreviation for organs to define concentration for the DDI matrix
LI = DEF.liver;                       DU = SYSTEM.CompNo + DEF.duodenum;
JE = SYSTEM.CompNo + DEF.jejunum;     IL = SYSTEM.CompNo + DEF.ileum;

%organs can be divided into three different subcompartments
cel = 2;

%intestinal segments can be divided into fluid, transit uptake and enterocytes
ent = 2;

%__DDI Matrix________________________________________________________________________
%%% Generate interaction parameters for all drugs %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Ki for CYP enzymes
Ki_CYP = permute( repmat( DDI.Ki_CYP(:, 1:DRUG.DrugNo), 1, 1, DDI.DDINo),...
                  [2, 3, 1]);

switch DRUG.DrugNo
    case 2
        %ddi_1 and ddi_2 are drug A and drug B alone
        Ki_CYP(drugB, ddi_1, :) = 1;    Ki_CYP(drugA, ddi_2, :) = 1;

    case 3
        %ddi_1, ddi_2 and ddi_3 are drug A, drug B and drug C alone
        Ki_CYP(drugB, ddi_1, :) = 1;    Ki_CYP(drugC, ddi_1, :) = 1;
        Ki_CYP(drugA, ddi_2, :) = 1;    Ki_CYP(drugC, ddi_2, :) = 1;
        Ki_CYP(drugA, ddi_3, :) = 1;    Ki_CYP(drugB, ddi_3, :) = 1;
end
```

```
%generate a factor to prevent dividing by 0 when Ki = 0
Fki_CYP = ones(DRUG.DrugNo, DDI.DDINo, SYSTEM.CYPliNo);
Fki_CYP(Ki_CYP == 1) = 0;

%MBI for CYP enzymes
kinact_CYP = permute( repmat( DDI.kinact_CYP(:, 1:DRUG.DrugNo),...
                              1, 1, DDI.DDINo), [2, 3, 1]);
Kapp_CYP   = permute( repmat( DDI.Kapp_CYP(:, 1:DRUG.DrugNo),...
                              1, 1, DDI.DDINo), [2, 3, 1]);
switch DRUG.DrugNo
    case 2
        kinact_CYP(drugB, ddi_1, :) = 0;     kinact_CYP(drugA, ddi_2, :) = 0;

        Kapp_CYP(drugB, ddi_1, :)   = 1;     Kapp_CYP(drugA, ddi_2, :)   = 1;
    case 3
        kinact_CYP(drugB, ddi_1, :) = 0;     kinact_CYP(drugC, ddi_1, :) = 0;
        kinact_CYP(drugA, ddi_2, :) = 0;     kinact_CYP(drugC, ddi_2, :) = 0;
        kinact_CYP(drugA, ddi_3, :) = 0;     kinact_CYP(drugB, ddi_3, :) = 0;

        Kapp_CYP(drugB, ddi_1, :)   = 1;     Kapp_CYP(drugC, ddi_1, :)   = 1;
        Kapp_CYP(drugA, ddi_2, :)   = 1;     Kapp_CYP(drugC, ddi_2, :)   = 1;
        Kapp_CYP(drugA, ddi_3, :)   = 1;     Kapp_CYP(drugB, ddi_3, :)   = 1;
end

%Induction for CYP enzymes
IndMax_CYP = permute( repmat( DDI.IndMax_CYP(:, 1:DRUG.DrugNo),...
                              1, 1, DDI.DDINo), [2, 3, 1]);
IC50_CYP   = permute( repmat( DDI.IC50_CYP(:, 1:DRUG.DrugNo),...
                              1, 1, DDI.DDINo), [2, 3, 1]);
switch DRUG.DrugNo
    case 2
        IndMax_CYP(drugB, ddi_1, :) = 0;     IndMax_CYP(drugA, ddi_2, :) = 0;

        IC50_CYP(drugB, ddi_1, :)   = 1;     IC50_CYP(drugA, ddi_2, :)   = 1;
    case 3
        IndMax_CYP(drugB, ddi_1, :) = 0;     IndMax_CYP(drugC, ddi_1, :) = 0;
        IndMax_CYP(drugA, ddi_2, :) = 0;     IndMax_CYP(drugC, ddi_2, :) = 0;
        IndMax_CYP(drugA, ddi_3, :) = 0;     IndMax_CYP(drugB, ddi_3, :) = 0;

        IC50_CYP(drugB, ddi_1, :)   = 1;     IC50_CYP(drugC, ddi_1, :)   = 1;
        IC50_CYP(drugA, ddi_2, :)   = 1;     IC50_CYP(drugC, ddi_2, :)   = 1;
        IC50_CYP(drugA, ddi_3, :)   = 1;     IC50_CYP(drugB, ddi_3, :)   = 1;
end

%%% choose the correct concentration and fu %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%multiplying factor for the concentration
Fco = ones(DRUG.DrugNo, DDI.DDINo);

switch DRUG.DrugNo
    case 2
        Fco(drugA, ddi_4) = 0;
        Fco(drugB, ddi_3) = 0;
    case 3
        Fco(drugA, ddi_5) = 0;    Fco(drugA, ddi_6) = 0;
        Fco(drugB, ddi_4) = 0;    Fco(drugB, ddi_6) = 0;
        Fco(drugC, ddi_4) = 0;    Fco(drugC, ddi_5) = 0;
end

%additive factor for the concentration depending on the compartment
FcelLI = zeros(DRUG.DrugNo, DDI.DDINo);
FentDU = zeros(DRUG.DrugNo, DDI.DDINo);
FentJE = zeros(DRUG.DrugNo, DDI.DDINo);
FentIL = zeros(DRUG.DrugNo, DDI.DDINo);

%fraction unbound for DDI predictions
fuLIcel = permute (repmat (DDI.fucel(:, DEF.liver, 1:DRUG.DrugNo), 1, DDI.DDINo, ↙
1),...
                   [3, 2, 1]);
fuGUent = permute (repmat (DDI.fucel(:, DEF.gut, :), 1, DRUG.DrugNo, 1),...
                   [2, 3, 1]);

switch DRUG.DrugNo
    case 2
        FcelLI(drugA, ddi_4) = LI + SYSTEM.SubNo(LI) + cel;
        FcelLI(drugB, ddi_3) = MODEL.ODENo + LI + SYSTEM.SubNo(LI) + cel;

        fuLIcel(drugA, ddi_2, :) = zeros(STUDY.IndNo, 1);
        fuLIcel(drugB, ddi_1, :) = zeros(STUDY.IndNo, 1);

        FentDU(drugA, ddi_4) = DU + SYSTEM.SubNo(DU) + ent;
        FentDU(drugB, ddi_3) = MODEL.ODENo + DU + SYSTEM.SubNo(DU) + ent;

        FentJE(drugA, ddi_4) = JE + SYSTEM.SubNo(JE) + ent;
        FentJE(drugB, ddi_3) = MODEL.ODENo + JE + SYSTEM.SubNo(JE) + ent;

        FentIL(drugA, ddi_4) = IL + SYSTEM.SubNo(IL) + ent;
        FentIL(drugB, ddi_3) = MODEL.ODENo + IL + SYSTEM.SubNo(IL) + ent;

        fuGUent(drugA, ddi_2, :) = zeros(STUDY.IndNo, 1);
        fuGUent(drugB, ddi_1, :) = zeros(STUDY.IndNo, 1);

    case 3
        FcelLI(drugA, [ddi_5, ddi_6]) = LI + SYSTEM.SubNo(LI) + cel;
        FcelLI(drugB, [ddi_4, ddi_6]) = MODEL.ODENo + LI + SYSTEM.SubNo(LI) + cel;
        FcelLI(drugC, [ddi_4, ddi_5]) = 2*MODEL.ODENo + LI + SYSTEM.SubNo(LI) + ↙
cel;

        fuLIcel(drugA, [ddi_2, ddi_3], :) = zeros(1, 2, STUDY.IndNo);
        fuLIcel(drugB, [ddi_1, ddi_3], :) = zeros(1, 2, STUDY.IndNo);
        fuLIcel(drugC, [ddi_1, ddi_2], :) = zeros(1, 2, STUDY.IndNo);

        FentDU(drugA, [ddi_5, ddi_6]) = DU + SYSTEM.SubNo(DU) + ent;
        FentDU(drugB, [ddi_4, ddi_6]) = MODEL.ODENo + DU + SYSTEM.SubNo(DU) + ent;
        FentDU(drugC, [ddi_4, ddi_5]) = 2*MODEL.ODENo + DU + SYSTEM.SubNo(DU) + ↙
ent;
```

```
        FentJE(drugA, [ddi_5, ddi_6]) = JE + SYSTEM.SubNo(JE) + ent;
        FentJE(drugB, [ddi_4, ddi_6]) = MODEL.ODENo + JE + SYSTEM.SubNo(JE) + ent;
        FentJE(drugC, [ddi_4, ddi_5]) = 2*MODEL.ODENo + JE + SYSTEM.SubNo(JE) + ↙
ent;

        FentIL(drugA, [ddi_5, ddi_6]) = IL + SYSTEM.SubNo(IL) + ent;
        FentIL(drugB, [ddi_4, ddi_6]) = MODEL.ODENo + IL + SYSTEM.SubNo(IL) + ent;
        FentIL(drugC, [ddi_4, ddi_5]) = 2*MODEL.ODENo + IL + SYSTEM.SubNo(IL) + ↙
ent;

        fuGUent(drugA, [ddi_2, ddi_3], :) = zeros(1, 2, STUDY.IndNo);
        fuGUent(drugB, [ddi_1, ddi_3], :) = zeros(1, 2, STUDY.IndNo);
        fuGUent(drugC, [ddi_1, ddi_2], :) = zeros(1, 2, STUDY.IndNo);
end

%%% Combine all data into one matrix to inform the ODE system %%%%%%%%%%%%%%%%%
STUDY.DDIMat = {'Fco', 'FcelLI', 'FentDU', 'FentJE', 'FentIL', 'fuLIcel',↙
'fuGUent';...
                Fco, FcelLI, FentDU, FentJE, FentIL, fuLIcel, fuGUent};

STUDY.DDIMat_CYP = {'Fki', 'Ki', 'kinact', 'Kapp', 'IndMax', 'IC50';...
                    Fki_CYP, Ki_CYP, kinact_CYP, Kapp_CYP, IndMax_CYP, IC50_CYP};

end

%================================================================================
%%% USED REFERENCES %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%================================================================================
% Sun D, Lennernäs H, Welage LS, Barnett JL, Landowski CP, Foster D, Fleisher D, ↙
Lee K-D, & Amidon GL. Comparison of human duodenum and Caco-2 gene expression ↙
profiles for 12,000 gene sequences tags and correlation with permeability of 26 ↙
drugs. Pharmaceutical Research, 2002. 19(10): 1400-1416.
% Rodgers T & Rowland M. Mechanistic approaches to volume of distribution ↙
predictions: understanding the processes. Pharmaceutical Research, 2007. 24(5): ↙
918-933.
% Poulin P & Theil FP. Prediction of pharmacokinetics prior to in vivo studies. 1. ↙
Mechanism?based prediction of volume of distribution. Journal of Pharmaceutical ↙
Sciences, 2002. 91(1): 129-156.
```

```
function[] = SolveODE()
%This function solves the ordinary differential equations

global DEF        %global DEF defines model parameters
global SYSTEM     %global SYSTEM defines sytem parameters
global DRUG       %global DRUG defines drug parameters
global DDI        %global DDI enhances drug parameters for DDI prediction
global STUDY      %global STUDY defines study design parameters
global MODEL      %global MODEL defines parameters important for modeling
global RES        %global RES saves the results for post-processing

%__Numbering variables for ODE solver___________________________________________
CompNo = SYSTEM.CompNo;                     %number of organs / tissues + blood
CoSeNo = SYSTEM.CompNo + SYSTEM.SegNo;      %compartments + intestinal segments
SubNo  = SYSTEM.SubNo;                      %number of subcompartments
IndNo  = STUDY.IndNo;                       %number of virtual individuals
DrugNo = DRUG.DrugNo;                       %number of drugs
DDINo  = DDI.DDINo;                         %number of DDI simulations
ODENo  = MODEL.ODENo;                       %number of ODE equations

%%% Model structure %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
LU = DEF.lung;                AD = DEF.adipose;               BO = DEF.bone;
BR = DEF.brain;               GO = DEF.gonads;                HE = DEF.heart;
KI = DEF.kidney;              MU = DEF.muscle;                SK = DEF.skin;
TH = DEF.thymus;              GU = DEF.gut;                   SP = DEF.spleen;
PA = DEF.pancreas;            LI = DEF.liver;                 LN = DEF.lymphnode;
RE = DEF.remaining;           VB = DEF.plasma;                AB = DEF.RBC;
ST = CompNo + DEF.stomach;    DU = CompNo + DEF.duodenum;
JE = CompNo + DEF.jejunum;    IL = CompNo + DEF.ileum;
CN = CompNo + DEF.colon;      FS = CompNo + DEF.faeces;

C2D6 = CoSeNo + DEF.CYP2D6;   C3A4 = CoSeNo + DEF.CYP3A4;
C3A5 = CoSeNo + DEF.CYP3A5;   C2J2 = CoSeNo + DEF.CYP2J2;

%%% No of equations %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if DRUG.DrugNo == 1
    NumEquations = ODENo * DrugNo;
else
    NumEquations = ODENo * DDINo;
end

%%% intestinal segments %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sto = DEF.stomach;            duo = DEF.duodenum;             jej = DEF.jejunum;
ile = DEF.ileum;              col = DEF.colon;

%%% CYP enzymes %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CYP2D6 = DEF.CYP2D6;          CYP3A4 = DEF.CYP3A4;
CYP3A5 = DEF.CYP3A5;          CYP2J2 = DEF.CYP2J2;

%__System data__________________________________________________________________
%%% Volumes %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Vvb  = SYSTEM.Vvein;      %volume for the venous blood pool
Vab  = SYSTEM.Vartery;    %volume for the arterial blood pool
```

```
Vvas = SYSTEM.Vvas;      %vascular volume for each compartment
Vint = SYSTEM.Vint;      %interstitial volume for each compartment
Vcel = SYSTEM.Vcel;      %intracellular volume for each compartment
Vlum = SYSTEM.VlumCAT;   %volume of the intestinal lumen
Vent = SYSTEM.VentCAT;   %volume of enterocytes in each segment of the intestine

%%% Blood and lymph flows %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CO   = SYSTEM.Qorg(:, LU);      %cardiac output
Qorg = SYSTEM.Qorg;             %regional blood flows
QHA  = SYSTEM.QHA;              %hepatic arterial blodo flow
QBY  = SYSTEM.QBY;              %blood flow of the liver bypass

Porg = Qorg.*(1-SYSTEM.HCT);    %plasma flow

Ltot = SYSTEM.TotLymphFlow ;    %total lymph flow
Lorg = SYSTEM.Lorg;             %regional lymph flows
QL   = Qorg - Lorg;             %substract regional blood from lymph flows
PL   = Porg - Lorg;             %substract regionalplasma from lymph flows

%%% GI tract %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ITT = 1./SYSTEM.TransitT;       %intestinal transit time

AbCYPin = SYSTEM.CYPseg_AB .* 10^3;    %abundance of intestinal enzymes in [pmol]
kdCYPin = SYSTEM.CYPin_kdeg;           %degradation rate of intestinal enzymes

%%% Liver %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WLI    = SYSTEM.Worg(:,LI);    %liver weight
MPPGL = SYSTEM.MPPGL;          %microsomal protein per gram liver (MPPGL)

AbCYPhe = SYSTEM.CYPhe_AB;     %hepatic CYP enzyme abundance
kdCYPhe = SYSTEM.CYPhe_kdeg;   %degradation rate of hepatic CYP enzymes

%__Drug data____________________________________________________________________
%%% PhysChem properties %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
MW    = DDI.MolW;      %molecular weight
fuine = DDI.fuine;     %fraction unbound in the interstitial space
fucel = DDI.fucel;     %fraction unbound in the intracellular space
BP    = DDI.BP;        %blood-to-plasma ratio

%%% Absorption %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CLab = DDI.CLab;       %absoprtion flux from the lumen into the enterocytes
LagR = DDI.LagRate;    %lag rate to delay Cmax/Tmax - parameter is artificial

%%% Distribution %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jin  = DDI.Jin;        %flux from the interstitial to the intracellular space
Jout = DDI.Jout;       %flux from the intracellular to the interstitial space

%%% Metabolism and Elimination %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CLint = DDI.CLint;     %intrinsic clearance of an unspecified enzyme pathway
CLbi  = DDI.CLbi;      %biliary clearance
CLre  = DDI.CLre;      %renal clearance
CLad  = DDI.CLad;      %additional plasma clearance
```

```
VmCYP = DDI.Vmax_CYP;     %Vmax for CYP enzymes
KmCYP = DDI.Km_CYP;       %KM for CYP enzymes
CiCYP = DDI.CLint_CYP;    %intrinisc clearance for CYP enzymes

%%% DDIs %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Fco = STUDY.DDIMat{2,1};    %factor for concentration
                            %1 = automatic concentration is used
                            %0 = concentration of the perpetrator is used

%factor for concentration of the perpetrator in the relevant compartment
FcelLI = STUDY.DDIMat{2,2};    %intracellular concentration in the liver
FentDU = STUDY.DDIMat{2,3};    %enterocytic concentration in the duodenum
FentJE = STUDY.DDIMat{2,4};    %enterocytic concentration in the jejunum
FentIL = STUDY.DDIMat{2,5};    %enterocytic concentration in the ileum

%fraction unbound ofr the perpetrator
fuLIcel = STUDY.DDIMat{2,6};    %fraction unbound in the liver
fuGUcel = STUDY.DDIMat{2,7};    %fraction unbound in the enterocytes

FKiCYP = STUDY.DDIMat_CYP{2,1};    %factor for competitive inhibition
                                   %0 = no competitive inhibition
                                   %1 = competitive inhibition is considered

KiCYP  = STUDY.DDIMat_CYP{2,2};    %inhibition constant

kinactCYP = STUDY.DDIMat_CYP{2,3};    %maximum inactivation rate constant
KappCYP   = STUDY.DDIMat_CYP{2,4};    %apparent enzyme inhibition constant
IndMaxCYP = STUDY.DDIMat_CYP{2,5};    %maximum fold of induction
IC50CYP   = STUDY.DDIMat_CYP{2,6};    %half-maximum induction

%__Study design_______________________________________________________________
Dose       = reshape(STUDY.DoseEventMat(:, 3, :), STUDY.NoEvents, DRUG.DrugNo);
AdminRoute = reshape(STUDY.DoseEventMat(:, 4, :), STUDY.NoEvents, DRUG.DrugNo);

rep = DDI.DDINo / DRUG.DrugNo;
Dose       = repmat(Dose, 1, rep);
AdminRoute = repmat(AdminRoute, 1, rep);

StartT = STUDY.DoseEventMat(:, 1, 1);    %start time for drug administration
EndT   = STUDY.DoseEventMat(:, 2, 1);    %end time for drug administration

NP = STUDY.DoseEventMat(:, 5, 1);              %resolution for one dosing event
NumPoints = sum(STUDY.DoseEventMat(:, 5, 1));  %resolution for entire simulation

%__Set up initial conditions__________________________________________________
%predefine matrices for the time and the concentration output
Conc = zeros(NumPoints, NumEquations*IndNo);

%set up the right concentration for multiple dosing
MultConc = [STUDY.DoseEventMat(:, 6, 1) + 1,...
            STUDY.DoseEventMat(:, 5, 1) + STUDY.DoseEventMat(:, 6, 1)];

disp('Start ODE');
```

```matlab
for ind = 1:IndNo
    %save time and concentration for each subject
    TInd = zeros(NumPoints, 1);
    CInd = zeros(NumPoints, NumEquations);

    %command to load a function to estimate the initial concentration C0
    C0 = Initialise_Conc();

        for ne = 1:STUDY.NoEvents
            %initial concentration for multiple dosing M0
            M0 = Initialise_Mult(C0);

%__Solve equations________________________________________________________________
            %use a stiff solver - alternatively ode45 might be used
            sol = ode15s(@rhs_function, [StartT(ne,1),EndT(ne,1)], M0);

%__Save solution to a vector______________________________________________________
            %time vector for each multiple dosing step is generated
            T = linspace(StartT(ne, 1), EndT(ne, 1), NP(ne, 1))';

            %evaluate the solution sol at each timepoint T
            C = deval(sol, T)';

            TInd(MultConc(ne, 1) : MultConc(ne,2), 1) = T;
            CInd(MultConc(ne, 1) : MultConc(ne,2), :) = C;

            %combine concentration for each individual
            for eq = 1:NumEquations
                Conc(:, (ind-1) * ODENo * DDINo + eq) = CInd(:, eq);
            end
            fprintf('No of event:        = %g\n',ne);
        end
        fprintf('No of subject:         = %g\n',ind);
end

%__Save solution of the ODE solver globally for post-processing__________________
RES.Time        = TInd;          %time in h
RES.Conc        = Conc;          %concentration in microM
MODEL.NP        = NP;
MODEL.NumPoints = NumPoints;

disp('End ODE');

%================================================================================
%%% USED FUNCTION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%================================================================================

%__Initialise conditions for single dose / first dose___________________________
function C0 = Initialise_Conc()

    %it is assumed that each concentration is 0
    C0 = zeros(1, NumEquations);
    for cdi = 1:DDINo
        %%% initial values for CYP abundance (necessary for MBI & induction) %%%
        c2D6 = ODENo*(cdi-1) + C2D6 + SubNo(C2D6);     %CYP2D6
        c3A4 = ODENo*(cdi-1) + C3A4 + SubNo(C3A4);     %CYP3A4
        c3A5 = ODENo*(cdi-1) + C3A5 + SubNo(C3A5);     %CYP3A5
        c2J2 = ODENo*(cdi-1) + C2J2 + SubNo(C2J2);     %CYP2J2

        %initial values for hepatic and intestinal CYP2D6
        C0(1, c2D6)    = AbCYPhe(ind, CYP2D6);
        C0(1, c2D6+1) = AbCYPin(ind, CYP2D6, duo);
        C0(1, c2D6+2) = AbCYPin(ind, CYP2D6, jej);
        C0(1, c2D6+3) = AbCYPin(ind, CYP2D6, ile);

        %initial values for hepatic and intestinal CYP3A4
        C0(1, c3A4)    = AbCYPhe(ind, CYP3A4);
        C0(1, c3A4+1) = AbCYPin(ind, CYP3A4, duo);
        C0(1, c3A4+2) = AbCYPin(ind, CYP3A4, jej);
        C0(1, c3A4+3) = AbCYPin(ind, CYP3A4, ile);

        %initial values for hepatic and intestinal CYP3A5
        C0(1, c3A5)    = AbCYPhe(ind, CYP3A5);
        C0(1,c3A5+1)   = AbCYPin(ind, CYP3A5, duo);
        C0(1,c3A5+2)   = AbCYPin(ind, CYP3A5, jej);
        C0(1,c3A5+3)   = AbCYPin(ind, CYP3A5, ile);

        %initial value for hepatic CYP2J2
        C0(1, c2J2)    = AbCYPhe(ind, CYP2J2);

        %%% initial values for drug concentration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        switch AdminRoute(1, cdi)

            %venous concentration in the case of intravenous administration
            case DEF.iv
                vb = ODENo*(cdi-1) + VB + SubNo(VB);
                C0(1, vb) = ((Dose(1, cdi) * 1000) / MW(cdi)) / Vvb(ind);

            %stomach concentration in teh case of oral drug administration
            case DEF.oral
                st = ODENo*(cdi-1) + ST + SubNo(ST);
                C0(1, st) = ((Dose(1,cdi) * 1000) / MW(cdi)) / Vlum(ind, sto);
        end
    end
end

%__Initialise conditions for multiple dosing____________________________________
function  M0 = Initialise_Mult(C0)

    %single or first dose
    if ne == 1
        %C0 has already been defined for a single / first dose
        M0 = C0;

    else
```

```
        %for all multiple doses, the concentration is defined by the solution sol
        M0 = CInd(MultConc(ne-1, 2), :);

        for mdi = 1:DDINo
            switch AdminRoute(ne,mdi)

                %venous concentration in the case of intravenous administration
                case DEF.iv
                    vb = ODENo*(mdi-1) + VB + SubNo(VB);
                    M0(1, vb) = (((Dose(ne, mdi) * 1000) / MW(mdi)) /...
                                Vvb(ind)) + CInd(MultConc(ne-1, 2), vb);

                %stomach concentration in teh case of oral drug administration
                case DEF.oral
                    st = ODENo*(mdi-1) + ST + SubNo(ST);
                    M0(1, st) = (((Dose(ne, mdi) * 1000) / MW(mdi)) /...
                                Vlum(ind, sto)) + CInd(MultConc(ne-1, 2), st);
            end
        end
    end
end

%__Define the right hand site of the equations__________________________________
function dtdy = rhs_function(~, y)

    %prepare the output as a column vector
    dtdy = zeros(NumEquations, 1);

    for d = 1:DDINo
        %define index for each compartment
        lu = ODENo*(d-1) + LU + SubNo(LU);     %lung
        ad = ODENo*(d-1) + AD + SubNo(AD);     %adipose
        bo = ODENo*(d-1) + BO + SubNo(BO);     %bone
        br = ODENo*(d-1) + BR + SubNo(BR);     %brain
        go = ODENo*(d-1) + GO + SubNo(GO);     %gonads
        he = ODENo*(d-1) + HE + SubNo(HE);     %heart
        ki = ODENo*(d-1) + KI + SubNo(KI);     %kidney
        mu = ODENo*(d-1) + MU + SubNo(MU);     %muscle
        sk = ODENo*(d-1) + SK + SubNo(SK);     %skin
        th = ODENo*(d-1) + TH + SubNo(TH);     %thymus
        gu = ODENo*(d-1) + GU + SubNo(GU);     %gut
        sp = ODENo*(d-1) + SP + SubNo(SP);     %spleen
        pa = ODENo*(d-1) + PA + SubNo(PA);     %pancreas
        li = ODENo*(d-1) + LI + SubNo(LI);     %liver
        ln = ODENo*(d-1) + LN + SubNo(LN);     %lymphnode
        re = ODENo*(d-1) + RE + SubNo(RE);     %remaining
        vb = ODENo*(d-1) + VB + SubNo(VB);     %venous
        ab = ODENo*(d-1) + AB + SubNo(AB);     %arterial

        st = ODENo*(d-1) + ST + SubNo(ST);     %stomach
        du = ODENo*(d-1) + DU + SubNo(DU);     %duodenum
        je = ODENo*(d-1) + JE + SubNo(JE);     %jejunum
        il = ODENo*(d-1) + IL + SubNo(IL);     %ileum
        cn = ODENo*(d-1) + CN + SubNo(CN);     %colon
        fs = ODENo*(d-1) + FS + SubNo(FS);     %faeces

        c2D6  = ODENo*(d-1) + C2D6 + SubNo(C2D6);     %CYP2D6
        c3A4  = ODENo*(d-1) + C3A4 + SubNo(C3A4);     %CYP3A4
        c3A5  = ODENo*(d-1) + C3A5 + SubNo(C3A5);     %CYP3A5
        c2J2  = ODENo*(d-1) + C2J2 + SubNo(C2J2);     %CYP2J2

        %define mechanism-based inhibition (MBI)
        MechLI = sum((kinactCYP(:, d, :) .* y((li+2)*Fco(:, d)+FcelLI(:, d)) .*↙
fuLIcel(:, d, ind)) ./ (KappCYP(:, d, :) + y((li+2)*Fco(:, d)+FcelLI(:, d)) .*↙
fuLIcel(:, d, ind)), 1);

        MechDU = sum((kinactCYP(:, d, :) .* y((du+2)*Fco(:, d)+FentDU(:, d)) .*↙
fuGUcel(:, d, ind)) ./ (KappCYP(:, d, :) + y((du+2)*Fco(:, d)+FentDU(:, d)) .*↙
fuGUcel(:, d, ind)), 1);
        MechJE = sum((kinactCYP(:, d, :) .* y((je+2)*Fco(:, d)+FentJE(:, d)) .*↙
fuGUcel(:, d, ind)) ./ (KappCYP(:, d, :) + y((je+2)*Fco(:, d)+FentJE(:, d)) .*↙
fuGUcel(:, d, ind)), 1);
        MechIL = sum((kinactCYP(:, d, :) .* y((il+2)*Fco(:, d)+FentIL(:, d)) .*↙
fuGUcel(:, d, ind)) ./ (KappCYP(:, d, :) + y((il+2)*Fco(:, d)+FentIL(:, d)) .*↙
fuGUcel(:, d, ind)), 1);

        %define induction
        InduLI = sum((IndMaxCYP(:, d, :) .* y((li+2)*Fco(:, d)+FcelLI(:, d)) .*↙
fuLIcel(:, d, ind)) ./ (IC50CYP(:, d, :) + y((li+2)*Fco(:, d)+FcelLI(:, d)) .*↙
fuLIcel(:, d, ind)), 1);

        InduDU = sum((IndMaxCYP(:, d, :) .* y((du+2)*Fco(:, d)+FentDU(:, d)) .*↙
fuGUcel(:, d, ind)) ./ (IC50CYP(:, d, :) + y((du+2)*Fco(:, d)+FentDU(:, d)) .*↙
fuGUcel(:, d, ind)), 1);
        InduJE = sum((IndMaxCYP(:, d, :) .* y((je+2)*Fco(:, d)+FentJE(:, d)) .*↙
fuGUcel(:, d, ind)) ./ (IC50CYP(:, d, :) + y((je+2)*Fco(:, d)+FentJE(:, d)) .*↙
fuGUcel(:, d, ind)), 1);
        InduIL = sum((IndMaxCYP(:, d, :) .* y((il+2)*Fco(:, d)+FentIL(:, d)) .*↙
fuGUcel(:, d, ind)) ./ (IC50CYP(:, d, :) + y((il+2)*Fco(:, d)+FentIL(:, d)) .*↙
fuGUcel(:, d, ind)), 1);

        %calculate the dynamic CYP abundance
        dtdy(c2D6)   = kdCYPhe(ind, CYP2D6) * AbCYPhe(ind, CYP2D6) * (1 + InduLI↙
(CYP2D6)) - (kdCYPhe(ind, CYP2D6) + MechLI(CYP2D6)) * y(c2D6);
        dtdy(c2D6+1) = kdCYPin(ind, CYP2D6) * AbCYPin(ind, CYP2D6, duo) * (1 +↙
InduDU(CYP2D6)) - (kdCYPin(ind, CYP2D6) + MechDU(CYP2D6)) * y(c2D6+1);
        dtdy(c2D6+2) = kdCYPin(ind, CYP2D6) * AbCYPin(ind, CYP2D6, jej) * (1 +↙
InduJE(CYP2D6)) - (kdCYPin(ind, CYP2D6) + MechJE(CYP2D6)) * y(c2D6+2);
        dtdy(c2D6+3) = kdCYPin(ind, CYP2D6) * AbCYPin(ind, CYP2D6, ile) * (1 +↙
InduIL(CYP2D6)) - (kdCYPin(ind, CYP2D6) + MechIL(CYP2D6)) * y(c2D6+3);

        dtdy(c3A4)   = kdCYPhe(ind, CYP3A4) * AbCYPhe(ind, CYP3A4) * (1 + InduLI↙
(CYP3A4)) - (kdCYPhe(ind, CYP3A4) + MechLI(CYP3A4)) * y(c3A4);
        dtdy(c3A4+1) = kdCYPin(ind, CYP3A4) * AbCYPin(ind, CYP3A4, duo) * (1 +↙
InduDU(CYP3A4)) - (kdCYPin(ind, CYP3A4) + MechDU(CYP3A4)) * y(c3A4+1);
        dtdy(c3A4+2) = kdCYPin(ind, CYP3A4) * AbCYPin(ind, CYP3A4, jej) * (1 +↙
InduJE(CYP3A4)) - (kdCYPin(ind, CYP3A4) + MechJE(CYP3A4)) * y(c3A4+2);
        dtdy(c3A4+3) = kdCYPin(ind, CYP3A4) * AbCYPin(ind, CYP3A4, ile) * (1 +↙
InduIL(CYP3A4)) - (kdCYPin(ind, CYP3A4) + MechIL(CYP3A4)) * y(c3A4+3);

        dtdy(c3A5)   = kdCYPhe(ind, CYP3A5) * AbCYPhe(ind, CYP3A5) * (1 + InduLI↙
(CYP3A5)) - (kdCYPhe(ind, CYP3A5) + MechLI(CYP3A5)) * y(c3A5);
        dtdy(c3A5+1) = kdCYPin(ind, CYP3A5) * AbCYPin(ind, CYP3A5, duo) * (1 +↙
InduDU(CYP3A5)) - (kdCYPin(ind, CYP3A5) + MechDU(CYP3A5)) * y(c3A5+1);
        dtdy(c3A5+2) = kdCYPin(ind, CYP3A5) * AbCYPin(ind, CYP3A5, jej) * (1 +↙
InduJE(CYP3A5)) - (kdCYPin(ind, CYP3A5) + MechJE(CYP3A5)) * y(c3A5+2);
        dtdy(c3A5+3) = kdCYPin(ind, CYP3A5) * AbCYPin(ind, CYP3A5, ile) * (1 +↙
InduIL(CYP3A5)) - (kdCYPin(ind, CYP3A5) + MechIL(CYP3A5)) * y(c3A5+3);

        dtdy(c2J2)   = kdCYPhe(ind, CYP2J2) * AbCYPhe(ind, CYP2J2) * (1 + InduLI↙
(CYP2J2)) - (kdCYPhe(ind, CYP2J2) + MechLI(CYP2J2)) * y(c2J2);

        %define competitive inhibition
        CYPComLI = reshape(1 + sum((((y((li+2)*Fco(:, d)+FcelLI(:, d)) .* fuLIcel↙
(:, d, ind) .* FKiCYP(:, d, :)) ./ KiCYP(:, d, :))), 1), SYSTEM.CYPliNo, 1);

        CYPComDU = reshape(1 + sum((((y((du+2)*Fco(:, d)+FentDU(:, d)) .* fuGUcel↙
(:, d, ind) .* FKiCYP(:, d, :)) ./ KiCYP(:, d, :))), 1), SYSTEM.CYPliNo, 1);
        CYPComJE = reshape(1 + sum((((y((je+2)*Fco(:, d)+FentJE(:, d)) .* fuGUcel↙
(:, d, ind) .* FKiCYP(:, d, :)) ./ KiCYP(:, d, :))), 1), SYSTEM.CYPliNo, 1);
        CYPComIL = reshape(1 + sum((((y((il+2)*Fco(:, d)+FentIL(:, d)) .* fuGUcel↙
(:, d, ind) .* FKiCYP(:, d, :)) ./ KiCYP(:, d, :))), 1), SYSTEM.CYPliNo, 1);

        %define enzymatic metbaolism
        CYPMetLI = (VmCYP(:, d) ./ ((KmCYP(:, d) .* CYPComLI) + y(li+2)*fucel(ind,↙
LI, d))) +...
                      (CiCYP(:, d) ./ CYPComLI);

        CYPMetDU = (VmCYP(:, d) ./ ((KmCYP(:, d) .* CYPComDU) + y(du+2)*fucel(ind,↙
GU, d))) +...
                      (CiCYP(:, d) ./ CYPComDU);
        CYPMetJE = (VmCYP(:, d) ./ ((KmCYP(:, d) .* CYPComJE) + y(je+2)*fucel(ind,↙
GU, d))) +...
                      (CiCYP(:, d) ./ CYPComJE);
        CYPMetIL = (VmCYP(:, d) ./ ((KmCYP(:, d) .* CYPComIL) + y(il+2)*fucel(ind,↙
GU, d))) +...
                      (CiCYP(:, d) ./ CYPComIL);

        %vascular space of compartments
        dtdy(lu)    = (1/Vvas(ind,LU)) * (Qorg(ind,LU)*y(vb) - QL(ind,LU)*y(lu) -↙
Porg(ind,LU)*(y(lu)/BP(d)) + PL(ind,LU)*y(lu+1));
        dtdy(ad)    = (1/Vvas(ind,AD)) * (Qorg(ind,AD)*y(ab) - QL(ind,AD)*y(ad) -↙
Porg(ind,AD)*(y(ad)/BP(d)) + PL(ind,AD)*y(ad+1));
        dtdy(bo)    = (1/Vvas(ind,BO)) * (Qorg(ind,BO)*y(ab) - QL(ind,BO)*y(bo) -↙
Porg(ind,BO)*(y(bo)/BP(d)) + PL(ind,BO)*y(bo+1));
        dtdy(br)    = (1/Vvas(ind,BR)) * (Qorg(ind,BR)*y(ab) - QL(ind,BR)*y(br) -↙
Porg(ind,BR)*(y(br)/BP(d)) + PL(ind,BR)*y(br+1));
        dtdy(go)    = (1/Vvas(ind,GO)) * (Qorg(ind,GO)*y(ab) - QL(ind,GO)*y(go) -↙
Porg(ind,GO)*(y(go)/BP(d)) + PL(ind,GO)*y(go+1));
```

```
        dtdy(he)    = (1/Vvas(ind,HE)) * (Qorg(ind,HE)*y(ab) - QL(ind,HE)*y(he) -↙
Porg(ind,HE)*(y(he)/BP(d)) + PL(ind,HE)*y(he+1));
        dtdy(ki)    = (1/Vvas(ind,KI)) * (Qorg(ind,KI)*y(ab) - QL(ind,KI)*y(ki) -↙
Porg(ind,KI)*(y(ki)/BP(d)) + PL(ind,KI)*y(ki+1) - CLre(ind,d)*y(ki));
        dtdy(mu)    = (1/Vvas(ind,MU)) * (Qorg(ind,MU)*y(ab) - QL(ind,MU)*y(mu) -↙
Porg(ind,MU)*(y(mu)/BP(d)) + PL(ind,MU)*y(mu+1));
        dtdy(sk)    = (1/Vvas(ind,SK)) * (Qorg(ind,SK)*y(ab) - QL(ind,SK)*y(sk) -↙
Porg(ind,SK)*(y(sk)/BP(d)) + PL(ind,SK)*y(sk+1));
        dtdy(th)    = (1/Vvas(ind,TH)) * (Qorg(ind,TH)*y(ab) - QL(ind,TH)*y(th) -↙
Porg(ind,TH)*(y(th)/BP(d)) + PL(ind,TH)*y(th+1));
        dtdy(gu)    = (1/Vvas(ind,GU)) * (Qorg(ind,GU)*y(ab) - QL(ind,GU)*y(gu) -↙
Porg(ind,GU)*(y(gu)/BP(d)) + PL(ind,GU)*y(gu+1));
        dtdy(sp)    = (1/Vvas(ind,SP)) * (Qorg(ind,SP)*y(ab) - QL(ind,SP)*y(sp) -↙
Porg(ind,SP)*(y(sp)/BP(d)) + PL(ind,SP)*y(sp+1));
        dtdy(pa)    = (1/Vvas(ind,PA)) * (Qorg(ind,PA)*y(ab) - QL(ind,PA)*y(pa) -↙
Porg(ind,PA)*(y(pa)/BP(d)) + PL(ind,PA)*y(pa+1));
        dtdy(li)    = (1/Vvas(ind,LI)) * (QHA(ind)*y(ab) + QBY(ind)*y(ab) + QL(ind,↙
GU)*y(gu) + QL(ind,SP)*y(sp) + QL(ind,PA)*y(pa) - QL(ind,LI)*y(li) - Porg(ind,LI)*↙
(y(li)/BP(d)) + PL(ind,LI)*y(li+1));
        dtdy(ln)    = (1/Vvas(ind,LN)) * (Qorg(ind,LN)*y(ab) - Qorg(ind,LN)*y(ln) +↙
Ltot(ind)*y(ln+1) - Ltot(ind)*y(ln));
        dtdy(re)    = (1/Vvas(ind,RE)) * (Qorg(ind,RE)*y(ab) - QL(ind,RE)*y(re) -↙
Porg(ind,RE)*(y(re)/BP(d)) + PL(ind,RE)*y(re+1));

        %interstitial space of compartments
        dtdy(lu+1) = (1/Vint(ind,LU)) * (Porg(ind,LU)*(y(lu)/BP(d)) - PL(ind,LU)*y↙
(lu+1) - Lorg(ind,LU)*y(lu+1) - Jin(ind,LU,d)*y(lu+1)*fuine(ind,LU,d) + Jout(ind,↙
LU,d)*y(lu+2)*fucel(ind,LU,d));
        dtdy(ad+1) = (1/Vint(ind,AD)) * (Porg(ind,AD)*(y(ad)/BP(d)) - PL(ind,AD)*y↙
(ad+1) - Lorg(ind,AD)*y(ad+1) - Jin(ind,AD,d)*y(ad+1)*fuine(ind,AD,d) + Jout(ind,↙
AD,d)*y(ad+2)*fucel(ind,AD,d));
        dtdy(bo+1) = (1/Vint(ind,BO)) * (Porg(ind,BO)*(y(bo)/BP(d)) - PL(ind,BO)*y↙
(bo+1) - Lorg(ind,BO)*y(bo+1) - Jin(ind,BO,d)*y(bo+1)*fuine(ind,BO,d) + Jout(ind,↙
BO,d)*y(bo+2)*fucel(ind,BO,d));
        dtdy(br+1) = (1/Vint(ind,BR)) * (Porg(ind,BR)*(y(br)/BP(d)) - PL(ind,BR)*y↙
(br+1) - Lorg(ind,BR)*y(br+1) - Jin(ind,BR,d)*y(br+1)*fuine(ind,BR,d) + Jout(ind,↙
BR,d)*y(br+2)*fucel(ind,BR,d));
        dtdy(go+1) = (1/Vint(ind,GO)) * (Porg(ind,GO)*(y(go)/BP(d)) - PL(ind,GO)*y↙
(go+1) - Lorg(ind,GO)*y(go+1) - Jin(ind,GO,d)*y(go+1)*fuine(ind,GO,d) + Jout(ind,↙
GO,d)*y(go+2)*fucel(ind,GO,d));
        dtdy(he+1) = (1/Vint(ind,HE)) * (Porg(ind,HE)*(y(he)/BP(d)) - PL(ind,HE)*y↙
(he+1) - Lorg(ind,HE)*y(he+1) - Jin(ind,HE,d)*y(he+1)*fuine(ind,HE,d) + Jout(ind,↙
HE,d)*y(he+2)*fucel(ind,HE,d));
        dtdy(ki+1) = (1/Vint(ind,KI)) * (Porg(ind,KI)*(y(ki)/BP(d)) - PL(ind,KI)*y↙
(ki+1) - Lorg(ind,KI)*y(ki+1) - Jin(ind,KI,d)*y(ki+1)*fuine(ind,KI,d) + Jout(ind,↙
KI,d)*y(ki+2)*fucel(ind,KI,d));
        dtdy(mu+1) = (1/Vint(ind,MU)) * (Porg(ind,MU)*(y(mu)/BP(d)) - PL(ind,MU)*y↙
(mu+1) - Lorg(ind,MU)*y(mu+1) - Jin(ind,MU,d)*y(mu+1)*fuine(ind,MU,d) + Jout(ind,↙
MU,d)*y(mu+2)*fucel(ind,MU,d));
        dtdy(sk+1) = (1/Vint(ind,SK)) * (Porg(ind,SK)*(y(sk)/BP(d)) - PL(ind,SK)*y↙
(sk+1) - Lorg(ind,SK)*y(sk+1) - Jin(ind,SK,d)*y(sk+1)*fuine(ind,SK,d) + Jout(ind,↙
SK,d)*y(sk+2)*fucel(ind,SK,d));
        dtdy(th+1) = (1/Vint(ind,TH)) * (Porg(ind,TH)*(y(th)/BP(d)) - PL(ind,TH)*y↙
(th+1) - Lorg(ind,TH)*y(th+1) - Jin(ind,TH,d)*y(th+1)*fuine(ind,TH,d) + Jout(ind, ↙
TH,d)*y(th+2)*fucel(ind,TH,d));
        dtdy(gu+1) = (1/Vint(ind,GU)) * (Porg(ind,GU)*(y(gu)/BP(d)) - PL(ind,GU)*y↙
(gu+1) - Lorg(ind,GU)*y(gu+1)+...
                                         Jin(ind,GU,d)*y(du+2)*fucel(ind,GU,d)+...
                                         Jin(ind,GU,d)*y(je+2)*fucel(ind,GU,d)+...
                                         Jin(ind,GU,d)*y(il+2)*fucel(ind,GU,d)+...
                                         Jin(ind,GU,d)*y(cn+2)*fucel(ind,GU,d));
        dtdy(sp+1) = (1/Vint(ind,SP)) * (Porg(ind,SP)*(y(sp)/BP(d)) - PL(ind,SP)*y↙
(sp+1) - Lorg(ind,SP)*y(sp+1) - Jin(ind,SP,d)*y(sp+1)*fuine(ind,SP,d) + Jout(ind, ↙
SP,d)*y(sp+2)*fucel(ind,SP,d));
        dtdy(pa+1) = (1/Vint(ind,PA)) * (Porg(ind,PA)*(y(pa)/BP(d)) - PL(ind,PA)*y↙
(pa+1) - Lorg(ind,PA)*y(pa+1) - Jin(ind,PA,d)*y(pa+1)*fuine(ind,PA,d) + Jout(ind, ↙
PA,d)*y(pa+2)*fucel(ind,PA,d));
        dtdy(li+1) = (1/Vint(ind,LI)) * (Porg(ind,LI)*(y(li)/BP(d)) - PL(ind,LI)*y↙
(li+1) - Lorg(ind,LI)*y(li+1) - Jin(ind,LI,d)*y(li+1)*fuine(ind,LI,d) + Jout(ind, ↙
LI,d)*y(li+2)*fucel(ind,LI,d));
        dtdy(ln+1) = (1/Vint(ind,LN)) * (Lorg(ind,AD)*y(ad+1) + Lorg(ind,BO)*y↙
(bo+1) + Lorg(ind,BR)*y(br+1) +...
                                         Lorg(ind,GO)*y(go+1) + Lorg(ind,HE)*y↙
(he+1) + Lorg(ind,KI)*y(ki+1) +...
                                         Lorg(ind,MU)*y(mu+1) + Lorg(ind,SK)*y↙
(sk+1) + Lorg(ind,TH)*y(th+1) +...
                                         Lorg(ind,GU)*y(gu+1) + Lorg(ind,SP)*y↙
(sp+1) + Lorg(ind,PA)*y(pa+1) +...
                                         Lorg(ind,LI)*y(li+1) + Lorg(ind,RE)*y↙
(re+1) + Lorg(ind,LU)*y(lu+1) - Ltot(ind)*y(ln+1)-...
                                         Jin(ind,LN,d)*y(ln+1)*fuine(ind,LN,d) +↙
Jout(ind,LN,d)*y(ln+2)*fucel(ind,LN,d));
        dtdy(re+1) = (1/Vint(ind,RE)) * (Porg(ind,RE)*(y(re)/BP(d)) - PL(ind,RE)*y↙
(re+1) - Lorg(ind,RE)*y(re+1) - Jin(ind,RE,d)*y(re+1)*fuine(ind,RE,d) + Jout(ind, ↙
RE,d)*y(re+2)*fucel(ind,RE,d));

        %intracellular space of compartments
        dtdy(lu+2) = (1/Vcel(ind,LU)) * (Jin(ind,LU,d)*y(lu+1)*fuine(ind,LU,d) -↙
Jout(ind,LU,d)*y(lu+2)*fucel(ind,LU,d));
        dtdy(ad+2) = (1/Vcel(ind,AD)) * (Jin(ind,AD,d)*y(ad+1)*fuine(ind,AD,d) -↙
Jout(ind,AD,d)*y(ad+2)*fucel(ind,AD,d));
        dtdy(bo+2) = (1/Vcel(ind,BO)) * (Jin(ind,BO,d)*y(bo+1)*fuine(ind,BO,d) -↙
Jout(ind,BO,d)*y(bo+2)*fucel(ind,BO,d));
        dtdy(br+2) = (1/Vcel(ind,BR)) * (Jin(ind,BR,d)*y(br+1)*fuine(ind,BR,d) -↙
Jout(ind,BR,d)*y(br+2)*fucel(ind,BR,d));
        dtdy(go+2) = (1/Vcel(ind,GO)) * (Jin(ind,GO,d)*y(go+1)*fuine(ind,GO,d) -↙
Jout(ind,GO,d)*y(go+2)*fucel(ind,GO,d));
        dtdy(he+2) = (1/Vcel(ind,HE)) * (Jin(ind,HE,d)*y(he+1)*fuine(ind,HE,d) -↙
Jout(ind,HE,d)*y(he+2)*fucel(ind,HE,d));
        dtdy(ki+2) = (1/Vcel(ind,KI)) * (Jin(ind,KI,d)*y(ki+1)*fuine(ind,KI,d) -↙
Jout(ind,KI,d)*y(ki+2)*fucel(ind,KI,d));
        dtdy(mu+2) = (1/Vcel(ind,MU)) * (Jin(ind,MU,d)*y(mu+1)*fuine(ind,MU,d) -↙
Jout(ind,MU,d)*y(mu+2)*fucel(ind,MU,d));
        dtdy(sk+2) = (1/Vcel(ind,SK)) * (Jin(ind,SK,d)*y(sk+1)*fuine(ind,SK,d) -↙
Jout(ind,SK,d)*y(sk+2)*fucel(ind,SK,d));
        dtdy(th+2) = (1/Vcel(ind,TH)) * (Jin(ind,TH,d)*y(th+1)*fuine(ind,TH,d) -↙
Jout(ind,TH,d)*y(th+2)*fucel(ind,TH,d));
        dtdy(sp+2) = (1/Vcel(ind,SP)) * (Jin(ind,SP,d)*y(sp+1)*fuine(ind,SP,d) -↙
Jout(ind,SP,d)*y(sp+2)*fucel(ind,SP,d));
        dtdy(pa+2) = (1/Vcel(ind,PA)) * (Jin(ind,PA,d)*y(pa+1)*fuine(ind,PA,d) -↙
Jout(ind,PA,d)*y(pa+2)*fucel(ind,PA,d));
        dtdy(li+2) = (1/Vcel(ind,LI)) * (Jin(ind,LI,d)*y(li+1)*fuine(ind,LI,d) -↙
Jout(ind,LI,d)*y(li+2)*fucel(ind,LI,d) -...
                                        (((CYPMetLI(CYP2D6)*y(c2D6)*MPPGL(ind)*WLI↙
(ind)*1000) +...
                                          (CYPMetLI(CYP3A4)*y(c3A4)*MPPGL(ind)*WLI↙
(ind)*1000) +...
                                          (CYPMetLI(CYP3A5)*y(c3A5)*MPPGL(ind)*WLI↙
(ind)*1000) +...
                                          (CYPMetLI(CYP2J2)*y(c2J2)*MPPGL(ind)*WLI↙
(ind)*1000) +...
                                          (CLint(d)*MPPGL(ind)*WLI(ind)*1000) +↙
CLbi(ind,d))*y(li+2)*fucel(ind,LI,d)));
        dtdy(ln+2) = (1/Vcel(ind,LN)) * (Jin(ind,LN,d)*y(ln+1)*fuine(ind,LN,d) -↙
Jout(ind,LN,d)*y(ln+2)*fucel(ind,LN,d));
        dtdy(re+2) = (1/Vcel(ind,RE)) * (Jin(ind,RE,d)*y(re+1)*fuine(ind,RE,d) -↙
Jout(ind,RE,d)*y(re+2)*fucel(ind,RE,d));

        %blood pools
        dtdy(vb)   = (1/Vvb(ind)) * (QL(ind,AD)*y(ad) + QL(ind,BO)*y(bo) + QL(ind,↙
BR)*y(br) + QL(ind,GO)*y(go) +...
                                     QL(ind,HE)*y(he) + QL(ind,KI)*y(ki) + QL(ind,↙
MU)*y(mu) + QL(ind,SK)*y(sk) +...
                                     QL(ind,TH)*y(th) + QL(ind,LI)*y(li) + Qorg↙
(ind,LN)*y(ln) + QL(ind,RE)*y(re) +...
                                     Ltot(ind)*y(ln) - CO(ind)*y(vb));

        dtdy(ab)   = (1/Vab(ind)) * (QL(ind,LU)*y(lu) - Qorg(ind,AD)*y(ab) - Qorg↙
(ind,BO)*y(ab) - Qorg(ind,BR)*y(ab) -...
                                     Qorg(ind,GO)*y(ab) - Qorg(ind,HE)*y(ab) - Qorg↙
(ind,KI)*y(ab) - Qorg(ind,MU)*y(ab) -...
                                     Qorg(ind,SK)*y(ab) - Qorg(ind,TH)*y(ab) - Qorg↙
(ind,GU)*y(ab) - Qorg(ind,SP)*y(ab) -...
                                     Qorg(ind,PA)*y(ab) - QHA(ind)*y(ab) - QBY(ind)↙
*y(ab) - Qorg(ind,LN)*y(ab) - Qorg(ind,RE)*y(ab) -...
                                     CLad(d)*y(ab));

        %intestinal lumen
        dtdy(st)   = (1/Vlum(ind,sto)) * (-ITT(ind,sto)*y(st)*Vlum(ind,sto));

        dtdy(du)   = (1/Vlum(ind,duo)) * (ITT(ind,sto)*y(st)*Vlum(ind,sto) - ITT↙
(ind,duo)*y(du)*Vlum(ind,duo) - CLab(ind,duo,d)*y(du));
        dtdy(je)   = (1/Vlum(ind,jej)) * (ITT(ind,duo)*y(du)*Vlum(ind,duo) - ITT↙
(ind,jej)*y(je)*Vlum(ind,jej) - CLab(ind,jej,d)*y(je));
        dtdy(il)   = (1/Vlum(ind,ile)) * (ITT(ind,jej)*y(je)*Vlum(ind,jej) - ITT↙
(ind,ile)*y(il)*Vlum(ind,ile) - CLab(ind,ile,d)*y(il));
        dtdy(cn)   = (1/Vlum(ind,col)) * (ITT(ind,ile)*y(il)*Vlum(ind,ile) - ITT↙
(ind,col)*y(cn)*Vlum(ind,col) - CLab(ind,col,d)*y(cn));
        dtdy(fs)   = ITT(ind,col)*y(cn)*Vlum(ind,col);
        %artificial uptake when a lag rate is necessary to delay Cmax / Tmax
        dtdy(du+1) = ((CLab(ind,duo,d)*y(du)) / Vlum(ind,duo) - LagR(d)*y(du+1));
        dtdy(je+1) = ((CLab(ind,jej,d)*y(je)) / Vlum(ind,jej) - LagR(d)*y(je+1));
        dtdy(il+1) = ((CLab(ind,ile,d)*y(il)) / Vlum(ind,ile) - LagR(d)*y(il+1));
        dtdy(cn+1) = ((CLab(ind,col,d)*y(cn)) / Vlum(ind,col) - LagR(d)*y(cn+1));

        %enetrocytes
        dtdy(du+2) = (1/Vent(ind,duo)) * (LagR(d)*y(du+1)*Vlum(ind,duo) - Jin(ind,↙
GU,d)*y(du+2)*fucel(ind,GU,d) -...
                                        ((CYPMetDU(CYP2D6)*y(c2D6+1) + CYPMetDU↙
(CYP3A4)*y(c3A4+1) + CYPMetDU(CYP3A5)*y(c3A5+1)) * y(du+2)*fucel(ind,GU,d)));
        dtdy(je+2) = (1/Vent(ind,jej)) * (LagR(d)*y(je+1)*Vlum(ind,jej) - Jin(ind,↙
GU,d)*y(je+2)*fucel(ind,GU,d) -...
                                        ((CYPMetJE(CYP2D6)*y(c2D6+2) + CYPMetJE↙
(CYP3A4)*y(c3A4+2) + CYPMetJE(CYP3A5)*y(c3A5+2)) * y(je+2)*fucel(ind,GU,d)));
        dtdy(il+2) = (1/Vent(ind,ile)) * (LagR(d)*y(il+1)*Vlum(ind,ile) - Jin(ind,↙
GU,d)*y(il+2)*fucel(ind,GU,d) -...
                                        ((CYPMetIL(CYP2D6)*y(c2D6+3) + CYPMetIL↙
(CYP3A4)*y(c3A4+3) + CYPMetIL(CYP3A5)*y(c3A5+3)) * y(il+2)*fucel(ind,GU,d)));
        dtdy(cn+2) = (1/Vent(ind,col)) * (LagR(d)*y(cn+1)*Vlum(ind,col) - Jin(ind,↙
GU,d)*y(cn+2)*fucel(ind,GU,d));
    end
end

end
```

```
function[] = PostProcessing()
%This function processes the data from the ODE solution and outputs the results
%Attention: Some statistical calculations (geomean, prctile) require
%the Statistical and Machine Learning toolbox

global DEF       %global DEF defines model parameters
global SYSTEM    %global SYSTEM defines sytem parameters
global DRUG      %global DRUG defines drug parameters
global DDI       %global DDI enhances drug parameters for DDI prediction
global STUDY     %global STUDY defines study design parameters
global MODEL     %global MODEL defines parameters important for modeling
global OBS       %global OBS saves observed parameters for the output
global RES       %global RES saves the results for post-processing
global MEAN      %global MEAN saves the mean of PK parameters
global STDV      %global STDV saves the standard deviation of PK parameters
global GEOM      %global GEOM saves the geometric mean of PK parameters
global PERC      %global PERC saves the percentiles of PK parameters

%post-processing is exemplarily shown for venous blood concentration

%variables used in this script
InDoEv = reshape(STUDY.DoseEventMat(:, 3, :), STUDY.NoEvents, DRUG.DrugNo);

IFD = zeros(1, DRUG.DrugNo);    %index for the first dosing event
ILD = zeros(1, DRUG.DrugNo);    %index for the last dosing event
for drug = 1:DRUG.DrugNo
    IFD(drug) = find(InDoEv(:, drug), 1, 'first');
    ILD(drug) = find(InDoEv(:, drug), 1, 'last');
end
IFD = repmat(IFD, 1, DDI.DDINo/DRUG.DrugNo);
ILD = repmat(ILD, 1, DDI.DDINo/DRUG.DrugNo);

Dose    = repmat(STUDY.Dose, STUDY.IndNo, DDI.DDINo/DRUG.DrugNo);

%__Extract the concentration from the ODE solution______________________________
%extract the venous blood concentration from the solution
VenousConc = zeros(MODEL.NumPoints, STUDY.IndNo, DDI.DDINo);

for d = 1:DDI.DDINo
    for ind = 1:STUDY.IndNo
        venous = (ind-1)*MODEL.ODENo*DDI.DDINo + MODEL.ODENo*(d-1) +...
                 DEF.plasma + SYSTEM.SubNo(DEF.plasma);
        VenousConc(:, ind, d) = RES.Conc(:, venous) .* DDI.MolW(d);
    end
end

%calculate statistics for venous blood concentration
VenousConcMean = mean(VenousConc, 2);
VenousConcPerc = prctile(VenousConc, [5, 95], 2);

%__Calculate PK parameters______________________________________________________
%Cmax / Tmax / AUCt
Venous_PKparaT = Calc_PKparaT(VenousConc, RES.Time);
```

```
%extract AUCt for the last dose event for the extrapolation of the AUCinf
VB_AUCtLast = zeros(STUDY.IndNo, DDI.DDINo);
for d = 1:DDI.DDINo
    VB_AUCtLast(:, d) = Venous_PKparaT{3} (:, ILD(d), d);
end

%%% Elimination rate and extrapolation of parameters %%%%%%%%%%%%%%%%%%%%%%%%
Venous_PKparaINF = Calc_PKparaINF(VenousConc, RES.Time, VB_AUCtLast);

%%% Statistics for PK parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Venous_PKparaSTAT = Calc_PKparaSTAT(Venous_PKparaT, Venous_PKparaINF);

for dn = 1:DDI.DDINo
    MEAN.Venous_CmaxFirst(:, dn) = Venous_PKparaSTAT{1,1} (:, IFD(dn), dn);
    MEAN.Venous_TmaxFirst(:, dn) = Venous_PKparaSTAT{2,1} (:, IFD(dn), dn);
    MEAN.Venous_AUCtFirst(:, dn) = Venous_PKparaSTAT{3,1} (:, IFD(dn), dn);

    MEAN.Venous_CmaxLast(:, dn)  = Venous_PKparaSTAT{1,1} (:, ILD(dn), dn);
    MEAN.Venous_TmaxLast(:, dn)  = Venous_PKparaSTAT{2,1} (:, ILD(dn), dn);
    MEAN.Venous_AUCtLast(:, dn)  = Venous_PKparaSTAT{3,1} (:, ILD(dn), dn);

    STDV.Venous_CmaxFirst(:, dn) = Venous_PKparaSTAT{1,2} (:, IFD(dn), dn);
    STDV.Venous_TmaxFirst(:, dn) = Venous_PKparaSTAT{2,2} (:, IFD(dn), dn);
    STDV.Venous_AUCtFirst(:, dn) = Venous_PKparaSTAT{3,2} (:, IFD(dn), dn);

    STDV.Venous_CmaxLast(:, dn)  = Venous_PKparaSTAT{1,2} (:, ILD(dn), dn);
    STDV.Venous_TmaxLast(:, dn)  = Venous_PKparaSTAT{2,2} (:, ILD(dn), dn);
    STDV.Venous_AUCtLast(:, dn)  = Venous_PKparaSTAT{3,2} (:, ILD(dn), dn);

    GEOM.Venous_CmaxFirst(:, dn) = Venous_PKparaSTAT{1,3} (:, IFD(dn), dn);
    GEOM.Venous_TmaxFirst(:, dn) = Venous_PKparaSTAT{2,3} (:, IFD(dn), dn);
    GEOM.Venous_AUCtFirst(:, dn) = Venous_PKparaSTAT{3,3} (:, IFD(dn), dn);

    GEOM.Venous_CmaxLast(:, dn)  = Venous_PKparaSTAT{1,3} (:, ILD(dn), dn);
    GEOM.Venous_TmaxLast(:, dn)  = Venous_PKparaSTAT{2,3} (:, ILD(dn), dn);
    GEOM.Venous_AUCtLast(:, dn)  = Venous_PKparaSTAT{3,3} (:, ILD(dn), dn);

    PERC.Venous_CmaxFirst(:, dn) = Venous_PKparaSTAT{1,4} (:, IFD(dn), dn);
    PERC.Venous_TmaxFirst(:, dn) = Venous_PKparaSTAT{2,4} (:, IFD(dn), dn);
    PERC.Venous_AUCtFirst(:, dn) = Venous_PKparaSTAT{3,4} (:, IFD(dn), dn);

    PERC.Venous_CmaxLast(:, dn)  = Venous_PKparaSTAT{1,4} (:, ILD(dn), dn);
    PERC.Venous_TmaxLast(:, dn)  = Venous_PKparaSTAT{2,4} (:, ILD(dn), dn);
    PERC.Venous_AUCtLast(:, dn)  = Venous_PKparaSTAT{3,4} (:, ILD(dn), dn);
end

MEAN.Venous_Thalf    = Venous_PKparaSTAT{4,1};
STDV.Venous_Thalf    = Venous_PKparaSTAT{4,2};
GEOM.Venous_Thalf    = Venous_PKparaSTAT{4,3};
PERC.Venous_Thalf    = Venous_PKparaSTAT{4,4};

MEAN.Venous_AUCinf   = Venous_PKparaSTAT{5,1};
```

```
STDV.Venous_AUCinf    = Venous_PKparaSTAT{5,2};
GEOM.Venous_AUCinf    = Venous_PKparaSTAT{5,3};
PERC.Venous_AUCinf    = Venous_PKparaSTAT{5,4};

MEAN.Venous_CLF       = Venous_PKparaSTAT{6,1};
STDV.Venous_CLF       = Venous_PKparaSTAT{6,2};
GEOM.Venous_CLF       = Venous_PKparaSTAT{6,3};
PERC.Venous_CLF       = Venous_PKparaSTAT{6,4};

MEAN.Venous_VDF       = Venous_PKparaSTAT{7,1};
STDV.Venous_VDF       = Venous_PKparaSTAT{7,2};
GEOM.Venous_VDF       = Venous_PKparaSTAT{7,3};
PERC.Venous_VDF       = Venous_PKparaSTAT{7,4};

MEAN.Venous_AUCratio = Venous_PKparaSTAT{8,1};
STDV.Venous_AUCratio = Venous_PKparaSTAT{8,2};
GEOM.Venous_AUCratio = Venous_PKparaSTAT{8,3};
PERC.Venous_AUCratio = Venous_PKparaSTAT{8,4};

%__output results_______________________________________________________________
%figure for concentration output
Create_Figure(RES.Time, VenousConcMean, VenousConcPerc);

%output parameters on screen
switch DRUG.DrugNo
    case 1
        d = 1;
        fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
        fprintf('Cmax [ng/mL]:          %g      %g      %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
        fprintf('Tmax [h]:              %g      %g      %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
        fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
        fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
        fprintf('CL [L/h]:              %g      %g      %g\n',    [MEAN.Venous_CLF↙
(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
        fprintf('VD [L/kg]:             %g      %g      %g\n\n', [MEAN.Venous_VDF↙
(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);

    case 2
        for d = 1:DDI.DDINo
            switch d
                case 1
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),     PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g       %g       %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),        PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g       %g       %g\n\n', [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),        PERC.Venous_VDF(3,d)]);

                case 2
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g       %g       %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g       %g       %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g       %g       %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g       %g       %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),     PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g       %g       %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),        PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g       %g       %g\n\n', [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),        PERC.Venous_VDF(3,d)]);

                case 3
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g       %g       %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g       %g       %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g       %g       %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g       %g       %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),     PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g       %g       %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),        PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g       %g       %g\n',    [MEAN.↙
Venous_VDF(d),      PERC.Venous_VDF(1,d),         PERC.Venous_VDF(3,d)]);
                    fprintf('AUC ratio              %g       %g       %g\n\n', [MEAN.↙
Venous_AUCratio(1),  PERC.Venous_AUCratio(1,1),  PERC.Venous_AUCratio(3,1)]);

                case 4
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g       %g       %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g       %g       %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g       %g       %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g       %g       %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),     PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g       %g       %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),        PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g       %g       %g\n',    [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),        PERC.Venous_VDF(3,d)]);
                    fprintf('AUC ratio              %g       %g       %g\n\n', [MEAN.↙
Venous_AUCratio(2),  PERC.Venous_AUCratio(1,2),  PERC.Venous_AUCratio(3,2)]);
            end
        end

    case 3
        for d = 1:DDI.DDINo
            switch d
                case 1
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',   [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',   [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',   [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',   [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g      %g      %g\n',   [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g      %g      %g\n\n', [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);

                case 2
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',   [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',   [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',   [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',   [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g      %g      %g\n',   [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g      %g      %g\n\n', [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);

                case 3
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',   [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',   [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',   [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',   [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g      %g      %g\n',   [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g      %g      %g\n\n', [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);

                case 4
```

```
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g      %g      %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g      %g      %g\n',    [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);
                    fprintf('AUC ratio              %g      %g      %g\n\n', [MEAN.↙
Venous_AUCratio(1),  PERC.Venous_AUCratio(1,1),  PERC.Venous_AUCratio(3,1)]);

                case 5
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g      %g      %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g      %g      %g\n',    [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);
                    fprintf('AUC ratio              %g      %g      %g\n\n', [MEAN.↙
Venous_AUCratio(2),  PERC.Venous_AUCratio(1,2),  PERC.Venous_AUCratio(3,2)]);

                case 6
                    fprintf('Calculated PK parameters for   %s:\n', DDI.Name{d});
                    fprintf('Cmax [ng/mL]:          %g      %g      %g\n',    [MEAN.↙
Venous_CmaxFirst(d), PERC.Venous_CmaxFirst(1,d), PERC.Venous_CmaxFirst(3,d)]);
                    fprintf('Tmax [h]:              %g      %g      %g\n',    [MEAN.↙
Venous_TmaxFirst(d), PERC.Venous_TmaxFirst(1,d), PERC.Venous_TmaxFirst(3,d)]);
                    fprintf('AUCt [ng*mL/h]:        %g      %g      %g\n',    [MEAN.↙
Venous_AUCtLast(d),  PERC.Venous_AUCtLast(1,d),  PERC.Venous_AUCtLast(3,d)]);
                    fprintf('AUCinf [ng*mL/h]:      %g      %g      %g\n',    [MEAN.↙
Venous_AUCinf(d),    PERC.Venous_AUCinf(1,d),    PERC.Venous_AUCinf(3,d)]);
                    fprintf('CL [L/h]:              %g      %g      %g\n',    [MEAN.↙
Venous_CLF(d),       PERC.Venous_CLF(1,d),       PERC.Venous_CLF(3,d)]);
                    fprintf('VD [L/kg]:             %g      %g      %g\n',    [MEAN.↙
Venous_VDF(d),       PERC.Venous_VDF(1,d),       PERC.Venous_VDF(3,d)]);
                    fprintf('AUC ratio              %g      %g      %g\n\n', [MEAN.↙
Venous_AUCratio(3),  PERC.Venous_AUCratio(1,3),  PERC.Venous_AUCratio(3,3)]);
            end
        end
end
```

```
%==============================================================================
%%% USED FUNCTION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%==============================================================================
%__Calculate PK parameters_____________________________________________________
%%% Cmax/Tmax/AUCt %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function PKparaT = Calc_PKparaT(Conc, Time)

    %prepare matrices for all calculated parameters
    Cmax = zeros(STUDY.IndNo, STUDY.NoEvents, DDI.DDINo);
    Tmax = zeros(STUDY.IndNo, STUDY.NoEvents, DDI.DDINo);
    AUCt = zeros(STUDY.IndNo, STUDY.NoEvents, DDI.DDINo);
    AUC  = zeros(MODEL.NumPoints, STUDY.IndNo, DDI.DDINo);

    %Time needs to be available for each individual and drug
    Time = repmat(Time, 1, STUDY.IndNo, DDI.DDINo);

    %point in time matrix to start and end each dosing event
    TimePointST = reshape(STUDY.DoseEventMat(:, 1, :) .* STUDY.Resolution,...
                          STUDY.NoEvents, DRUG.DrugNo);
    TimePointEN = reshape(STUDY.DoseEventMat(:, 2, :) .* STUDY.Resolution,...
                          STUDY.NoEvents, DRUG.DrugNo);

    %the first index for the start point cannot be 0, but needs to be 1
    TimePointST(1, :, :) = 1;

    TimePointST = repmat(TimePointST, 1, DDI.DDINo / DRUG.DrugNo);
    TimePointEN = repmat(TimePointEN, 1, DDI.DDINo / DRUG.DrugNo);

    %Extract Cmax and Tmax for each dosing event and calculate AUCt
    for sub = 1:STUDY.IndNo

        for dr = 1:DDI.DDINo

            for ev = 1:STUDY.NoEvents

                %Calculation of Cmax / Tmax are only done if a dose is given
                if TimePointST(ev, dr) ~= 0
                    for t = TimePointST(ev, dr) : TimePointEN(ev, dr)
                        if Conc(t, sub, dr) > Cmax(sub, ev, dr)
                            Cmax(sub, ev, dr) = Conc(t, sub, dr);
                            Tmax(sub, ev, dr) = Time(t, sub, dr);
                        end

                        %trapezoidal method to calculate the AUC
                        if t == TimePointST(ev, dr)
                            AUC(t, sub, dr) = 0;
                        else
                            AUC(t, sub, dr) = ((Conc(t,   sub, dr) +...
                                                Conc(t-1, sub, dr)) .*...
                                               (Time(t,   sub, dr) -...
                                                Time(t-1, sub, dr)) ./ 2) +...
                                              AUC(t-1, sub, dr);
                        end
                    end
                end

                if TimePointEN(ev, dr) ~= 0
                    AUCt(sub, ev, dr) = AUC(TimePointEN(ev, dr), sub, dr);
                end

            end

        end

    end

    %save all parameters in one cell array
    PKparaT = {Cmax, Tmax, AUCt};
    end

    %%% extrapolate parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    function PKparaINF = Calc_PKparaINF(Conc, Time, AUCt)

        %take the log of the concentration
        LogConc = log10(Conc);

        %calculate the slope and beta for the last 5 time points
        slope = zeros(STUDY.IndNo, 2, DDI.DDINo);
        beta = zeros(STUDY.IndNo, DDI.DDINo);

        for dr = 1:DDI.DDINo
        for sub = 1:STUDY.IndNo
            slope(sub, :, dr) = polyfit(Time(end-4:end),...
                                        LogConc(end-4:end, sub, dr), 1);
            beta(sub, dr) = abs(slope(sub, 1, dr));
        end
        end

        %calculate the half-life and extrapolate the AUC to infitity
        Thalf   = log(2) ./ beta;
        AUCinf = AUCt + (reshape(Conc(end, :, :), STUDY.IndNo, DDI.DDINo) ./ beta);

        %calculate the AUC ratio for DDI prediction
        AUCratio = ones(STUDY.IndNo, DRUG.DrugNo);
        if DRUG.DrugNo == 1
        AUCratio(:, 1) = 1.0;

        else
        for de = 1:DRUG.DrugNo
            AUCratio(:, de) = AUCinf(:, de + DRUG.DrugNo) ./ AUCinf(:, de);
        end
        end

        CLF = (Dose.*1000) ./ AUCinf;
        VDF = (CLF ./ beta) ./ SYSTEM.Weight;
        %save all results in one cell array
        PKparaINF = {Thalf, AUCinf, CLF, VDF, AUCratio};
    end

    %%% calculate statistics of PK parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    function PKparaSTAT = Calc_PKparaSTAT(ParTIME, ParINF)

        per = [5, 50, 95];

        Mean_Cmax  = mean(ParTIME{1}, 1);       Geom_Cmax  = geomean(ParTIME{1}, 1);
        SD_Cmax    = std(ParTIME{1}, '', 1);    Perc_Cmax  = prctile(ParTIME{1}, per, ↙
    1);

        Mean_Tmax  = mean(ParTIME{2}, 1);       Geom_Tmax  = geomean(ParTIME{2}, 1);
        SD_Tmax    = std(ParTIME{2}, '', 1);    Perc_Tmax  = prctile(ParTIME{2}, per, ↙
    1);

        Mean_AUCt  = mean(ParTIME{3}, 1);       Geom_AUCt  = geomean(ParTIME{3}, 1);
        SD_AUCt    = std(ParTIME{3}, '', 1);    Perc_AUCt  = prctile(ParTIME{3}, per, ↙
    1);

        Mean_Thalf = mean(ParINF{1}, 1);       Geom_Thalf = geomean(ParINF{1}, 1);
        SD_Thalf   = std(ParINF{1}, '', 1);     Perc_Thalf = prctile(ParINF{1}, per, 1);

        Mean_AUCi  = mean(ParINF{2}, 1);       Geom_AUCi  = geomean(ParINF{2}, 1);
        SD_AUCi    = std(ParINF{2}, '', 1);     Perc_AUCi  = prctile(ParINF{2}, per, 1);

        Mean_CLF   = mean(ParINF{3}, 1);       Geom_CLF   = geomean(ParINF{3}, 1);
        SD_CLF     = std(ParINF{3}, '', 1);     Perc_CLF   = prctile(ParINF{3}, per, 1);

        Mean_VDF   = mean(ParINF{4}, 1);        Geom_VDF   = geomean(ParINF{4}, 1);
        SD_VDF     = std(ParINF{4}, '', 1);     Perc_VDF   = prctile(ParINF{4}, per, 1);

        Mean_AUCra = mean(ParINF{5}, 1);        Geom_AUCra = geomean(ParINF{5}, 1);
        SD_AUCra   = std(ParINF{5}, '', 1);     Perc_AUCra = prctile(ParINF{5}, per, 1);

        %save results
        PKparaSTAT = {Mean_Cmax,  SD_Cmax,  Geom_Cmax,  Perc_Cmax;...
                      Mean_Tmax,  SD_Tmax,  Geom_Tmax,  Perc_Tmax;...
                      Mean_AUCt,  SD_AUCt,  Geom_AUCt,  Perc_AUCt;...
                      Mean_Thalf, SD_Thalf, Geom_Thalf, Perc_Thalf;...
                      Mean_AUCi,  SD_AUCi,  Geom_AUCi,  Perc_AUCi;...
                      Mean_CLF,   SD_CLF,   Geom_CLF,   Perc_CLF;...
                      Mean_VDF,   SD_VDF,   Geom_VDF,   Perc_VDF;...
                      Mean_AUCra, SD_AUCra, Geom_AUCra, Perc_AUCra};
    end

    %__Output plasma concentration__________________________________________________
    function FigPlot = Create_Figure(Time, Mean, Perc)

        %prepare simulation time for the visualisation of the 95% CI
        DDIplotT = [Time', fliplr(Time')];
        DDIplotL = [Time(2:MODEL.NumPoints)', fliplr(Time(2:MODEL.NumPoints)')];
        switch DRUG.DrugNo

            case 1
                Plot1 = figure;

                %first subplot shows concentration
                subplot(1,2,1); hold on;
                %draw area between percentiles
                Plot1t = area(Time, [Perc(:,1,1), Perc(:,2,1) - Perc(:,1,1)],↙
    'LineStyle', 'none');
                %the area between x.axis and 5% CI should be white
                Plot1t(1).FaceColor = [1 1 1];
                %the area between 5 and 95% CI gets a light green
                Plot1t(2).FaceColor = [0.88 0.94 0.85];
                plot(Time, Mean(:,1), '-k',  'LineWidth', 1.5);
                errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, 'or',↙
    'LineWidth', 1.5);
                xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                ylabel('Venous Conc. [ng/mL]', 'fontweight', 'bold', 'fontsize', 12);
                set(gca, 'fontsize', 12);

                %second subplot shows concentration on a log scale
                subplot(1,2,2); hold on;
                %draw area between percentiles
                Plot1t = area(Time, [Perc(:,1,1), Perc(:,2,1) - Perc(:,1,1)],↙
    'LineStyle', 'none');
                %the area between x.axis and 5% CI should be white
                Plot1t(1).FaceColor = [1 1 1];
                %the area between 5 and 95% CI gets a light green
                Plot1t(2).FaceColor = [0.88 0.94 0.85];
                plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,1), 'k',↙
    'LineWidth', 1.5);
                errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, 'or',↙
    'LineWidth', 1.5);
                set(gca, 'fontsize', 12);
                set(gca, 'Yscale', 'log');
                set(gca, 'ycolor', 'k');
                set(gca, 'xcolor', 'k');
                xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold', 'fontsize', 12);
                text(-0.3,1.05, DDI.Name{1}, 'Units', 'normalized', 'fontweight',↙
    'bold' , 'fontsize', 14);
                set(Plot1, 'Units', 'normalized', 'Position', [0.2 0.2 0.6 0.6]);

            case 2
                for dr = 1:DDI.DDINo
                    switch dr

                        case 1
                            Plot1 = figure;

                            %first subplot shows concentration
                            subplot(1,2,1); hold on;
                            %draw area between percentiles
                            Plot1t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
    dr)], 'LineStyle', 'none');
                            %the area between x.axis and 5% CI should be white
                            Plot1t(1).FaceColor = [1 1 1];
                            %the area between 5 and 95% CI gets a light green
                            Plot1t(2).FaceColor = [0.88 0.94 0.85];
                            plot(Time, Mean(:,dr), '-k',  'LineWidth', 1.5);
                            errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
    'or', 'LineWidth', 1.5);
                            xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                            ylabel('Venous Conc. [ng/mL]', 'fontweight', 'bold',↙
    'fontsize', 12);
                            set(gca, 'fontsize', 12);

                            %second subplot shows concentration on a log scale
                            subplot(1,2,2); hold on;
                            %draw area between percentiles
                            Plot1t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
    dr)], 'LineStyle', 'none');
                            %the area between x.axis and 5% CI should be white
                            Plot1t(1).FaceColor = [1 1 1];
                            %the area between 5 and 95% CI gets a light green
                            Plot1t(2).FaceColor = [0.88 0.94 0.85];
                            plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,dr), ↙
    'k', 'LineWidth', 1.5);
                            errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
    'or', 'LineWidth', 1.5);
                            set(gca, 'fontsize', 12);
                            set(gca, 'Yscale', 'log');
                            set(gca, 'ycolor', 'k');
                            set(gca, 'xcolor', 'k');
                            xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                            ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
    'fontsize', 12);
                            text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
    'fontweight', 'bold' , 'fontsize', 14);
                            set(Plot1, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
    0.6]);

                        case 2
                            Plot2 = figure;

                            %first subplot shows concentration
                            subplot(1,2,1); hold on;
                            %draw area between percentiles
                            Plot2t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
    dr)], 'LineStyle', 'none');
                            %the area between x.axis and 5% CI should be white
                            Plot2t(1).FaceColor = [1 1 1];
                            %the area between 5 and 95% CI gets a light green
                            Plot2t(2).FaceColor = [0.88 0.94 0.85];
                            plot(Time, Mean(:,dr), '-k',  'LineWidth', 1.5);
                            errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2, ↙
    'or', 'LineWidth', 1.5);
                            xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                            ylabel('Venous Conc. [ng/mL]', 'fontweight', 'bold',↙
    'fontsize', 12);
                            set(gca, 'fontsize', 12);

                            %second subplot shows concentration on a log scale
                            subplot(1,2,2); hold on;
                            %draw area between percentiles
                            Plot2t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
    dr)], 'LineStyle', 'none');
                            %the area between x.axis and 5% CI should be white
                            Plot2t(1).FaceColor = [1 1 1];
                            %the area between 5 and 95% CI gets a light green
                            Plot2t(2).FaceColor = [0.88 0.94 0.85];
                            plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,dr), ↙
    'k', 'LineWidth', 1.5);
                            errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2, ↙
    'or', 'LineWidth', 1.5);
                            set(gca, 'fontsize', 12);
                            set(gca, 'Yscale', 'log');
                            set(gca, 'ycolor', 'k');
                            set(gca, 'xcolor', 'k');
                            xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                            ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
    'fontsize', 12);
                            text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
    'fontweight', 'bold' , 'fontsize', 14);
                            set(Plot2, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
    0.6]);

                        case 3
                            Plot3 = figure;

                            %first subplot shows concentration
                            subplot(1,2,1); hold on;

                            %draw area between percentiles
                            DDI1_area_Subst = [Perc(:,1,1)', fliplr(Perc(:,2,1)')];
                            DDI1_area_Perp  = [Perc(:,1,3)', fliplr(Perc(:,2,3)')];
                            DDI1_area_SuPe  = [Perc(:,2,1)', fliplr(Perc(:,1,3)')];

                            %fill different areas with different colours
                            fill(DDIplotT, DDI1_area_Subst, [0.88 0.94 0.85], ↙
    'Edgecolor', 'none');
                            fill(DDIplotT, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
    'none');
                            h = fill(DDIplotT, DDI1_area_SuPe, [0.35 0.63 0], ↙
    'Edgecolor', 'none');
                            set(h,'facealpha',0.1)
                            %plot the concentrations
                            plot(Time, Mean(:,1), 'k', 'LineWidth', 1.5);
                            plot(Time, Mean(:,3), '--b', 'LineWidth', 1.5);
                            errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
    'or', 'LineWidth', 1.5);
                            errorbar(OBS.Time_DDI1, OBS.Conc_DDI1, OBS.SD_DDI1, '+r',↙
    'LineWidth', 1.5);
                            xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                            ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
    'fontsize', 12);
                            set(gca, 'fontsize', 12);

                            %second subplot is in log scale
                            subplot(1,2,2); hold on;

                            %draw area between percentiles
                            DDI1_area_Subst = [Perc(2:MODEL.NumPoints,1,1)', fliplr↙
    (Perc(2:MODEL.NumPoints,2,1)')];
                            DDI1_area_Perp  = [Perc(2:MODEL.NumPoints,1,3)', fliplr↙
    (Perc(2:MODEL.NumPoints,2,3)')];
                            DDI1_area_SuPe  = [Perc(2:MODEL.NumPoints,2,1)', fliplr↙
    (Perc(2:MODEL.NumPoints,1,3)')];

                            %fill different areas with different colours
                            fill(DDIplotL, DDI1_area_Subst, [0.88 0.94 0.85],↙
    'Edgecolor', 'none');
                            fill(DDIplotL, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
    'none');
                            hlog = fill(DDIplotL, DDI1_area_SuPe, [0.35 0.63 0],↙
    'Edgecolor', 'none');
                            set(hlog,'facealpha',0.1)

                            %plot the concentrations
                            plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,1),↙
    'k', 'LineWidth', 1.5);
                            plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,3),↙
    '--b', 'LineWidth', 1.5);
                            errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
    'or', 'LineWidth', 1.5);
                            errorbar(OBS.Time_DDI1, OBS.Conc_DDI1, OBS.SD_DDI1, '+r',↙
    'LineWidth', 1.5);
                            set(gca, 'fontsize', 12);
                            set(gca, 'Yscale', 'log');
                            set(gca, 'ycolor', 'k');
                            set(gca, 'xcolor', 'k');
                            xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                            ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
    'fontsize', 12);
                            text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
    'fontweight', 'bold' , 'fontsize', 14);
                            set(Plot3, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
    0.6]);
```

```
                    case 4
                        Plot4 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(:,1,2)', fliplr(Perc(:,2,2)')];
                        DDI1_area_Perp  = [Perc(:,1,4)', fliplr(Perc(:,2,4)')];
                        DDI1_area_SuPe  = [Perc(:,2,2)', fliplr(Perc(:,1,4)')];

                        %fill different areas with different colours
                        fill(DDIplotT, DDI1_area_Subst, [0.88 0.94 0.85],↙
'Edgecolor', 'none');
                        fill(DDIplotT, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        h = fill(DDIplotT, DDI1_area_SuPe, [0.35 0.63 0],↙
'Edgecolor', 'none');
                        set(h,'facealpha',0.1)

                        %plot the concentrations
                        plot(Time, Mean(:,2), 'k', 'LineWidth', 1.5);
                        plot(Time, Mean(:,4), '--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2,↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI2, OBS.Conc_DDI2, OBS.SD_DDI2, '+r',↙
'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);

                        %second subplot is in log scale
                        subplot(1,2,2); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(2:MODEL.NumPoints,1,2)', fliplr↙
(Perc(2:MODEL.NumPoints,2,2)')];
                        DDI1_area_Perp  = [Perc(2:MODEL.NumPoints,1,4)', fliplr↙
(Perc(2:MODEL.NumPoints,2,4)')];
                        DDI1_area_SuPe  = [Perc(2:MODEL.NumPoints,2,2)', fliplr↙
(Perc(2:MODEL.NumPoints,1,4)')];

                        %fill different areas with different colours
                        fill(DDIplotL, DDI1_area_Subst, [0.88 0.94 0.85],↙
'Edgecolor', 'none');
                        fill(DDIplotL, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        hlog = fill(DDIplotL, DDI1_area_SuPe, [0.35 0.63 0],↙
'Edgecolor', 'none');
                        set(hlog,'facealpha',0.1)

                        %plot the concentrations
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,2), ↙
'k', 'LineWidth', 1.5);
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,4), ↙
'--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2, ↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI2, OBS.Conc_DDI2, OBS.SD_DDI2,  '+r',↙
'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Plot4, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
0.6]);
                end
            end

        case 3
            for dr = 1:DDI.DDINo
                switch dr

                    case 1
                        Plot1 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;
                        %draw area between percentiles
                        Plot1t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
dr)], 'LineStyle', 'none');
                        %the area between x.axis and 5% CI should be white
                        Plot1t(1).FaceColor = [1 1 1];
                        %the area between 5 and 95% CI gets a light green
                        Plot1t(2).FaceColor = [0.88 0.94 0.85];
                        plot(Time, Mean(:,dr), '-k',  'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
'or', 'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Venous Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);

                        %second subplot shows concentration on a log scale
                        subplot(1,2,2); hold on;
                        %draw area between percentiles
                        Plot1t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
dr)], 'LineStyle', 'none');
                        %the area between x.axis and 5% CI should be white
                        Plot1t(1).FaceColor = [1 1 1];
                        %the area between 5 and 95% CI gets a light green
                        Plot1t(2).FaceColor = [0.88 0.94 0.85];
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,dr),↙
'k', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
'or', 'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Plot1, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
0.6]);

                    case 2
                        Plot2 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;
                        %draw area between percentiles
                        Plot2t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
dr)], 'LineStyle', 'none');
                        %the area between x.axis and 5% CI should be white
                        Plot2t(1).FaceColor = [1 1 1];
                        %the area between 5 and 95% CI gets a light green
                        Plot2t(2).FaceColor = [0.88 0.94 0.85];
                        plot(Time, Mean(:,dr), '-k',  'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2, ↙
'or', 'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Venous Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);

                        %second subplot shows concentration on a log scale
                        subplot(1,2,2); hold on;
                        %draw area between percentiles
                        Plot2t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
dr)], 'LineStyle', 'none');
                        %the area between x.axis and 5% CI should be white
                        Plot2t(1).FaceColor = [1 1 1];
                        %the area between 5 and 95% CI gets a light green
                        Plot2t(2).FaceColor = [0.88 0.94 0.85];
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,dr),↙
'k', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2, ↙
'or', 'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Plot2, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
0.6]);

                    case 3
                        Plot3 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;
                        %draw area between percentiles
                        Plot3t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
dr)], 'LineStyle', 'none');
                        %the area between x.axis and 5% CI should be white
                        Plot3t(1).FaceColor = [1 1 1];
                        %the area between 5 and 95% CI gets a light green
                        Plot3t(2).FaceColor = [0.88 0.94 0.85];
                        plot(Time, Mean(:,dr), '-k',  'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug3, OBS.Conc_Drug3, OBS.SD_Drug3, ↙
'or', 'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Venous Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);

                        %second subplot shows concentration on a log scale
                        subplot(1,2,2); hold on;
                        %draw area between percentiles
                        Plot3t = area(Time, [Perc(:,1,dr), Perc(:,2,dr) - Perc(:,1,↙
dr)], 'LineStyle', 'none');
                        %the area between x.axis and 5% CI should be white
                        Plot3t(1).FaceColor = [1 1 1];
                        %the area between 5 and 95% CI gets a light green
                        Plot3t(2).FaceColor = [0.88 0.94 0.85];
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,dr), ↙
'k', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug3, OBS.Conc_Drug3, OBS.SD_Drug3, ↙
'or', 'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Plot3, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
0.6]);

                    case 4
                        Plot4 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(:,1,1)', fliplr(Perc(:,2,1)')];
                        DDI1_area_Perp  = [Perc(:,1,4)', fliplr(Perc(:,2,4)')];
                        DDI1_area_SuPe  = [Perc(:,2,1)', fliplr(Perc(:,1,4)')];

                        %fill the area between percentiles
                        fill(DDIplotT, DDI1_area_Subst, [0.88 0.94 0.85], ↙
'Edgecolor', 'none');
                        fill(DDIplotT, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        h = fill(DDIplotT, DDI1_area_SuPe, [0.35 0.63 0], ↙
'Edgecolor', 'none');
                        set(h,'facealpha',0.1)

                        %plot the concentration
                        plot(Time, Mean(:,1), 'k', 'LineWidth', 1.5);
                        plot(Time, Mean(:,4), '--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI1, OBS.Conc_DDI1, OBS.SD_DDI1, '+r',↙
'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);

                        %second subplot shows concentration on a log scale
                        subplot(1,2,2); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(2:MODEL.NumPoints,1,1)', fliplr↙
(Perc(2:MODEL.NumPoints,2,1)')];
                        DDI1_area_Perp  = [Perc(2:MODEL.NumPoints,1,4)', fliplr↙
(Perc(2:MODEL.NumPoints,2,4)')];
                        DDI1_area_SuPe  = [Perc(2:MODEL.NumPoints,2,1)', fliplr↙
(Perc(2:MODEL.NumPoints,1,4)')];

                        %fill the area between percentiles
                        fill(DDIplotL, DDI1_area_Subst, [0.88 0.94 0.85], ↙
'Edgecolor', 'none');
                        fill(DDIplotL, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        hlog = fill(DDIplotL, DDI1_area_SuPe, [0.35 0.63 0],↙
'Edgecolor', 'none');
                        set(hlog,'facealpha',0.1)
                        %plot the concentration
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,1), ↙
'k', 'LineWidth', 1.5);
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,4), ↙
'--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug1, OBS.Conc_Drug1, OBS.SD_Drug1, ↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI1, OBS.Conc_DDI1, OBS.SD_DDI1,  '+r',↙
'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Plot4, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
0.6]);

                    case 5
                        Plot5 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(:,1,2)', fliplr(Perc(:,2,2)')];
                        DDI1_area_Perp  = [Perc(:,1,5)', fliplr(Perc(:,2,5)')];
                        DDI1_area_SuPe  = [Perc(:,2,2)', fliplr(Perc(:,1,5)')];

                        %fill the area between percentiles
                        fill(DDIplotT, DDI1_area_Subst, [0.88 0.94 0.85], ↙
'Edgecolor', 'none');
                        fill(DDIplotT, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        h = fill(DDIplotT, DDI1_area_SuPe, [0.35 0.63 0], ↙
'Edgecolor', 'none');
                        set(h,'facealpha',0.1)

                        %plot the concentration
                        plot(Time, Mean(:,2), 'b', 'LineWidth', 1.5);
                        plot(Time, Mean(:,5), '--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2, ↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI2, OBS.Conc_DDI2, OBS.SD_DDI2,  '+r',↙
'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);
                        %second subplot shows concentration on a log scale
                        subplot(1,2,2); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(2:MODEL.NumPoints,1,2)', fliplr↙
(Perc(2:MODEL.NumPoints,2,2)')];
                        DDI1_area_Perp  = [Perc(2:MODEL.NumPoints,1,5)', fliplr↙
(Perc(2:MODEL.NumPoints,2,5)')];
                        DDI1_area_SuPe  = [Perc(2:MODEL.NumPoints,2,2)', fliplr↙
(Perc(2:MODEL.NumPoints,1,5)')];

                        %fill the area between percentiles
                        fill(DDIplotL, DDI1_area_Subst, [0.88 0.94 0.85],↙
'Edgecolor', 'none');
                        fill(DDIplotL, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        hlog = fill(DDIplotL, DDI1_area_SuPe, [0.35 0.63 0],↙
'Edgecolor', 'none');
                        set(hlog,'facealpha',0.1)

                        %plot the concentration
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,2),↙
'b', 'LineWidth', 1.5);
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,5),↙
'--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug2, OBS.Conc_Drug2, OBS.SD_Drug2,↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI2, OBS.Conc_DDI2, OBS.SD_DDI2, '+r',↙
'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Plot5, 'Units', 'normalized', 'Position', [0.2 0.2 0.6↙
0.6]);

                    case 6
                        Venous6 = figure;

                        %first subplot shows concentration
                        subplot(1,2,1); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(:,1,3)', fliplr(Perc(:,2,3)')];
                        DDI1_area_Perp  = [Perc(:,1,6)', fliplr(Perc(:,2,6)')];
                        DDI1_area_SuPe  = [Perc(:,2,3)', fliplr(Perc(:,1,6)')];
                        %fill the area between percentiles
                        fill(DDIplotT, DDI1_area_Subst, [0.88 0.94 0.85],↙
'Edgecolor', 'none');
                        fill(DDIplotT, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        h = fill(DDIplotT, DDI1_area_SuPe, [0.35 0.63 0],↙
'Edgecolor', 'none');
                        set(h,'facealpha',0.1)

                        %plot the concentration
                        plot(Time, Mean(:,3), 'b', 'LineWidth', 1.5);
                        plot(Time, Mean(:,6), '--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug3, OBS.Conc_Drug3, OBS.SD_Drug3,↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI3, OBS.Conc_DDI3, OBS.SD_DDI3, '+r',↙
'LineWidth', 1.5);
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        set(gca, 'fontsize', 12);

                        %second subplot shows concentration on a log scale
                        subplot(1,2,2); hold on;

                        %draw area between percentiles
                        DDI1_area_Subst = [Perc(2:MODEL.NumPoints,1,3)', fliplr↙
(Perc(2:MODEL.NumPoints,2,3)')];
                        DDI1_area_Perp  = [Perc(2:MODEL.NumPoints,1,6)', fliplr↙
(Perc(2:MODEL.NumPoints,2,3)')];
                        DDI1_area_SuPe  = [Perc(2:MODEL.NumPoints,2,3)', fliplr↙
(Perc(2:MODEL.NumPoints,1,6)')];

                        %fill the area between percentiles
                        fill(DDIplotL, DDI1_area_Subst, [0.88 0.94 0.85],↙
'Edgecolor', 'none');
                        fill(DDIplotL, DDI1_area_Perp, [0.81 0.92 1], 'Edgecolor',↙
'none');
                        hlog = fill(DDIplotL, DDI1_area_SuPe, [0.35 0.63 0],↙
'Edgecolor', 'none');
                        set(hlog,'facealpha',0.1)

                        %plot the concentration
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,3),↙
'b', 'LineWidth', 1.5);
                        plot(Time(2:MODEL.NumPoints), Mean(2:MODEL.NumPoints,6),↙
'--b', 'LineWidth', 1.5);
                        errorbar(OBS.Time_Drug3, OBS.Conc_Drug3, OBS.SD_Drug3,↙
'or', 'LineWidth', 1.5);
                        errorbar(OBS.Time_DDI3, OBS.Conc_DDI3, OBS.SD_DDI3, '+r',↙
'LineWidth', 1.5);
                        set(gca, 'fontsize', 12);
                        set(gca, 'Yscale', 'log');
                        set(gca, 'ycolor', 'k');
                        set(gca, 'xcolor', 'k');
                        xlabel('Time [h]', 'fontweight', 'bold', 'fontsize', 12);
                        ylabel('Plasma Conc. [ng/mL]', 'fontweight', 'bold',↙
'fontsize', 12);
                        text(-0.3,1.05, DDI.Name{dr}, 'Units', 'normalized',↙
'fontweight', 'bold' , 'fontsize', 14);
                        set(Venous6, 'Units', 'normalized', 'Position', [0.2 0.2↙
0.6 0.6]);
                end
            end
    end
FigPlot = 1;
end

end
```