This document describe some of the computation included in the paper:

Optimality for models given by a system of ordinary differential equation

Last update: 2014-01-10

Authors: Juan M. Rodríguez-Díaz and Guillermo Sánchez-León
Department of Statistics, Faculty of Science, Pl. de los Caídos s/n, 37008 Salamanca, Spain
ENUSA Industrias Avanzadas S.A. Salamanca, Spain

Many processes are given by a system of ordinary differential equations, very often without an analytical solution. When there are unknown parameters, that need to be estimated, optimum experimental design approach offers quality estimators for the different objectives of the practitioners. But almost every optimality criteria needs to deal with the linearized model for computing optimal designs, and this can be a great problem when it is not possible to obtain the analytical form of the model. In this work, a procedure for findingoptimal designs for models given as solutions to a system of ordinary differential equations is described. Some important models like the compartmental one, are studied through actual case studies, obtaining the corresponding optimal designs.

In[1]:=
```
$Version
```

Out[1]=
```
9.0 for Microsoft Windows (64-bit) (January 25, 2013)
```

Overview of compartmental and biokinectic equivalents models

Compartmental analysis has applications in clinical medicine, pharmacokinetics, internal dosimetry, nuclear medicine, ecosystem studies and chemical reaction kinetics. It can be described as the analysis of a system in terms of compartments which separate the system into a finite number of component parts which are called compartments. Compartments interact through the exchange of species. Species may be a chemical substance, hormone, individuals in a population and so on. A compartmental system is usually represented by a flow diagram or a block diagram. A general introduction to this theory can be found in Anderson (1983), Godfrey (1983) and Jazquez (1985).

We adopt the convention of representing compartments with circles or rectangles. The flow into or out of the compartments is represented by arrows. The i^{th} compartment of a system of n compartments is labelled i and the size (amount or content) of the component in compartment i as $x_i(t)$. The exchange between compartments, or between a compartment and the environment is labeled k_{ij}, where i represents the flow from i to j. The environment is usually represented by "0" (zero), so k_{i0} is the fractional excretion coefficient from the i-th compartment to the outside environment. The input from the environment into the j-th compartment is called $b_j(t)$. Environment represents the processes that are outside the system. With regards to the environment, we only need to know the flow, $b_j(t)$, into the system from the outside. The k_{ij} are called fractional transfer rate coefficients and they may be a function of different variables or constants.

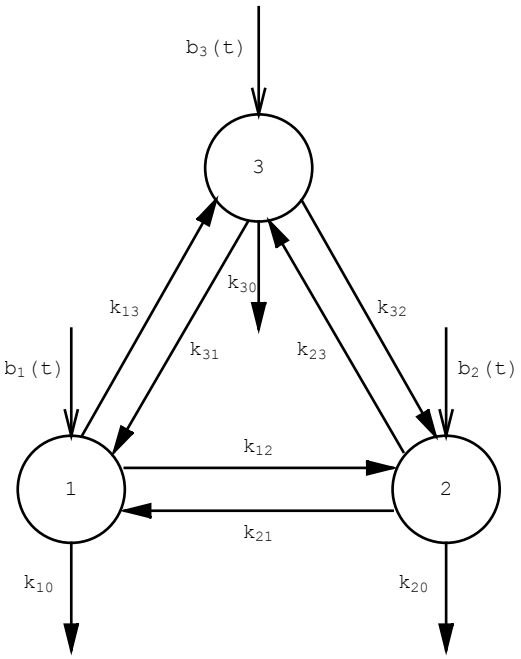


Figure 1. The general tricompartmental system.

Figure 1 represents the general tricompartmental model with one input and output in each compartment. If we suppose that the substance introduced into the system is a radioactive isotope we mush considered the radioactive decay, it is given by a constant rate represented by λ (decay constant), it is specific for each isotope (obviously $\lambda = 0$ if no radioactive substances are presents or they are long life isotopes) . The decay constant can be interpreted by an equal flow going out of the system in each compartment. The model can be represented by the follow ODE System

$$x_1'(t) = b_1(t) - (k_{10} + k_{12} + k_{13} + \lambda)\, x_1(t) + k_{21}\, x_2(t) + k_{31}\, x_3(t)$$
$$x_2'(t) = b_2(t) + k_{12}\, x_1(t) - (k_{20} + k_{21} + k_{23} + \lambda)\, x_2(t) + k_{32}\, x_3(t)$$
$$x_3'(t) = b_3(t) + k_{13}\, x_1(t) + k_{23}\, x_2(t) - (k_{30} + k_{31} + k_{32} + \lambda)\, x_3(t)$$

in matrix notation

$$\begin{pmatrix} x_1'(t) \\ x_2'(t) \\ x_2'(t) \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} b_1(t) \\ b_2(t) \\ b_3(t) \end{pmatrix}$$

with

$$a_{11} = -(k_{10} + k_{12} + k_{13} + \lambda);\ a_{12} = k_{21};\ a_{13} = k_{31};$$
$$a_{21} = k_{12};\ a_{22} = -(k_{20} + k_{21} + k_{23} + \lambda);\ a_{23} = k_{32};$$
$$a_{31} = k_{13};\ a_{32} = k_{23};\ a_{33} = -(k_{30} + k_{31} + k_{32} + \lambda);$$

In some circumstances, instead of compartments we have parts o process than can be represented by transference between Different parts, for example, when blood flow or other physiological parameter is measured. In those circumstances the transfer rate constants k_{ij} are associated with physiologically meaningful values that correspond to the measured physiological parameter or may be a function of them. However, if the physiological parameters are constants models are mathematically equivalent with compartmental models. We will refer in general to kinetic models.

The patterns that we have seen can be expanded to systems of n compartments or n state variables (in the case of physiological models). The equation for any compartment i in notation matrix is given by

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\boldsymbol{x}(t) + \mathbf{b}(t),\ t \geq 0$$
$$\mathbf{x}(0) = \mathbf{x}_0 \qquad (1)$$

where:

$\mathbf{x}(t) = \{x_1(t),\ x_2(t),\ \ ...,\ x_n(t)\}^T$ is a column vector and $x_i(t)$ denote the amount or content of species in compartment i at time t.

$\mathbf{A}$ is a $n \times n$ is usually known as the system matrix given by coefficients a_{ij}= g(k_{ij}) where k_{ij} are constant specifics of each model (in compartmental models they are obtained in the form that has been descried)

$\mathbf{b}(t) = \{b_1(t),\ b_2(t),\ ...,\ b_n(t)\}^T$ is a column vector where $\{b_i(t)\}$ is the input rate into compartment i from outside system.

$\mathbf{x}(0) = \{x_1(0),\ x_2(0),\ \ ...,\ x_n(0)\}^T$ are the initial conditions, so $x_i(0)$, represents the amount or content of species in compartment i at time $t = 0$.

The solution of eq(1) when the coeffs a_{ij} are constants is eq(2)

$$\boldsymbol{x}(t) = \mathbf{x}_0 \,\text{Exp}(\mathbf{A}\,t) + \text{Exp}(\mathbf{A}\,t) * \mathbf{b}(t) \tag{2}$$

where * denotes convolution,

$$\text{Exp}(\mathbf{A}\,t) * \mathbf{b}(t) = \int_0^t \text{Exp}[(t-\tau)\,\mathbf{A}]\,\mathbf{b}(\tau)\,d\tau$$

Sometime Laplace transforms are used to solve eq (1).

$$X(s) = (s\,I - A)^{-1}\,x_0 + (s\,I - A)^{-1}\,B(s) \tag{3}$$

where $X(s)$ and $B(s)$ are the Laplace transforms of $x(t)$ and $b(t)$. Then evaluating the inverse Laplace transformation is obtained:

$$\boldsymbol{x}(t) = \mathcal{L}^{-1}\left((s\,I - A)^{-1}\,x_0\right) + \mathcal{L}^{-1}\left((s\,I - A)^{-1}\,B(s)\right) \tag{4}$$

Both methods are used for BIOKMOD software for solving eq.(1) developped for ones of the authors.

In many circumstance some parameters of biokinetic models represented by eq. (1) (one o more k_{ij} coefficients), that we will call $\beta = \{\beta_1,\ ...,\ \beta_p\}$, are unknown an they are estimated by fitting experimental data. It means that we measure $y_i(t)$, being $y_i(t) = x_i(t, \beta)+\epsilon$, for different moments of t: $\hat{y}_i(t_0)$, ..., $\hat{y}_i(t_n)$. Then we estimated $\beta = \{\beta_1,\ ...,\ \beta_p\}$ by fitting $y_i(t)$ and $\hat{y}_i(t)$.

We propose chose the best moments $\{t_0,..,\ t_i,\ ...\ t_n\}$ to take the experimental data. It it can be done using techniques of Optimal Design of Experiment (ODE), in particular we chose apply a D-optimal design.

In a D-optimal design the values of $\{t_1,..,\ t_i,\ ...\ t_n\}$ (t_i is the time when the i-th sample should be taken) are given in the points that leads the determinant of the Fisher information matrix (M) to a maximum. The process to obtain M will be described later (it can be found in Hill, P. D. H., 1980, D-optimal designs for partially nonlinear regression models. Technometrics 22:275–276).

Derivatives

The standard method to obtain the maximum of Det|M| requires to know the analytical expression of $x_i(t, \beta)$. In this paper we propose a method to compute the D-optimal design in biokinetic system descried by ODE with the pattern of eq (2) when analytical expression of $x_i(t, \beta)$ cannot be obtained.

In the D-optimal design method the derivatives with respect to β must be evaluated

We used a subscript in parentheses to denote differentiation with respect to a parameter, so $\partial x / \partial \beta = x_{(\beta)}$.

When in eq(1) $\mathbf{A}$ depends of β, but not $\mathbf{b}(t)$ and $\mathbf{x}_0$ we can get the derivatives of $x_i(t, \beta)$ by differentiation of eq.(1)

$$\dot{\mathbf{x}}_{(p)}(t) = \mathbf{A}\,\boldsymbol{x}_{(p)}(t) + \mathbf{A}_{(p)}\,\boldsymbol{x}(t) \tag{5}$$

Then, using eq(2)

$$\boldsymbol{x}_{(p)}(t) = \text{Exp}(\mathbf{A}\,t)\ \mathbf{x}_{(p)}(0) + \text{Exp}(\mathbf{A}\,t) * \left[\mathbf{A}_{(p)}\ \boldsymbol{x}(t)\right] =$$
$$\text{Exp}(\mathbf{A}\,t)\ \mathbf{x}_{(p)}(0) + \text{Exp}(\mathbf{A}\,t) * \left[\mathbf{A}_{(p)}\ \ \text{Exp}(\mathbf{A}\,t)\,\mathbf{x}_0\right] + \text{Exp}(\mathbf{A}\,t) * \mathbf{A}_{(p)}\,\text{Exp}(\mathbf{A}\,t) * \mathbf{b}(t)$$

$$\boldsymbol{x}_{(p)}(t) = \text{Exp}(\mathbf{A}\,t)\ \mathbf{x}_{(p)}(0) + \mathbf{A}_{(p)}\,\mathbf{x}_0\,\text{Exp}\,(\mathbf{A}\,t) * \text{Exp}(\mathbf{A}\,t) + \mathbf{A}_{(p)}\,\text{Exp}(\mathbf{A}\,t) * \text{Exp}(\mathbf{A}\,t) * \mathbf{b}(t) \tag{6}$$

As $\boldsymbol{x}(0) = \boldsymbol{x}_0 \Rightarrow \mathbf{x}_{(p)}(0)=0$ therefore:

$$\boldsymbol{x}_{(p)}(t) = \mathbf{A}_{(p)}\,\mathbf{x}_0\,\text{Exp}\,(\mathbf{A}\,t) * \text{Exp}(\mathbf{A}\,t) + \mathbf{A}_{(p)}\,\text{Exp}(\mathbf{A}\,t) * \text{Exp}(\mathbf{A}\,t) * \mathbf{b}(t) \tag{7}$$

In the the particular case that the input happen in $t = 0$, that is $\mathbf{x}(0) = \mathbf{x}_0$ with $\mathbf{b}(t)=0$ for $t > 0$. Then

$$\boldsymbol{x}_{(p)}(t) = \mathbf{A}_{(p)}\,\mathbf{x}_0\,\text{Exp}\,(\mathbf{A}\,t) * \text{Exp}(\mathbf{A}\,t) \tag{8}$$

$$\text{Exp}(\mathbf{A}\,t) * \text{Exp}(\mathbf{A}\,t) \;=\; \int_0^t \text{Exp}[(t-\tau)\,\mathbf{A}]\,\text{Exp}(\mathbf{A}\,\tau)\,d\tau$$

The iodine model

The model

Let's consider the iodine biokinetic model represented in the figure 2 (ICRP 78) where compartment 1 is the blood, compartment 2 is the thyroid, compartment 3 is the rest of the body, compartment 4 is the bladder, $3 \to 0$, i.e. a transfer from compartment 3 to the environment, represents the output to the gastro intestinal tract (GIT) and $4 \to 0$ represents the output, via urine excretion, to the environment. In the follow we won`t consider compartment 4 that is not relevant in our case. We will assume a flow from compartment 1 to outside given by a transfer coefficients k_{10}. The coefficient transfer values, in days^{-1}, taken from ICRP 78 are k_{10}= 1.9404, k_{30}= 0.01155 and k_{31}= 0.0462. We will suppose that k_{12} and k_{23} are unknown, although we know that their values will be about k_{12} = 0.8 and k_{23} = 0.0078. We wish estimate them taken experiment data from compartment 1 . The problem consist on decide by DOE the best moment to taken the sample.

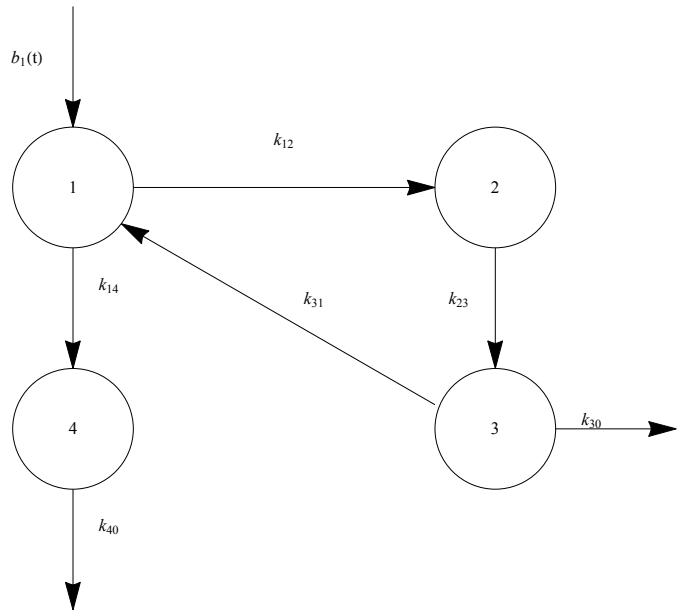


Figure 2. Iodine biokinetic models (ICRP 78)

Here it is downloaded the package Sysmodel (included in the Biokmod Tool available in http://www3.enusa.es/webMathematica/Public/Docs/biokmod.zip)

In[2]:=
```
Needs["Biokmod`SysModel`"]
```

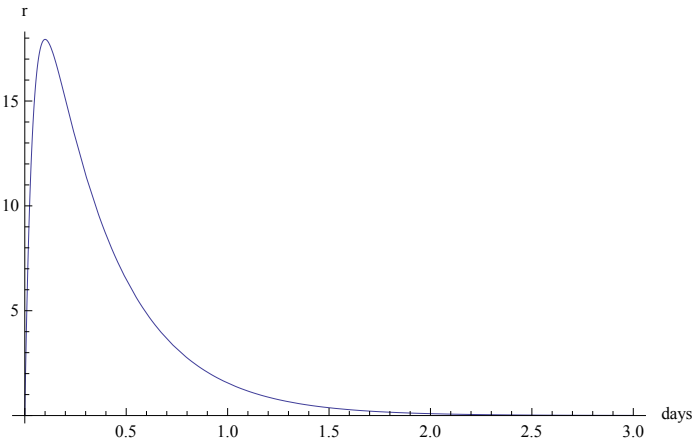
```
SysModel, version 1.5.1   2013-11-12
```

In this example it is assumed an input into b_1(t) compartment 1 is given by

In[3]:=
```
B = {-27.13 e^-24.08 t + 27.13 e^-2.86 t - 0.020 e^-0.147 t + 0.0194 e^-0.093 t, 0, 0};
```

In[4]:=
```
Plot[B[[1]], {t, 0, 3}, AxesLabel → {"days", "r"}]
```

Out[4]=



This kind of input happens in real situations when there is an input from the GIT (Gastro Intestinal) to the blood, for instance if the iodine is intaken by orally. Then b_1(t) represents the input from GIT to blood. (When the input happens by inhalation the flow to the blood can be represented by $34.4\, e^{-200.\, t} + 1.09\, e^{-110.\, t} + 0.808\, e^{-102.\, t} + 6.414\, e^{-100.\, t} + 5.458\, e^{-24.\, t}$ that is the function used in the article). The initial condition are { 0, 0, 0}.

In this kind of experiment usually is used a isotope of iodine with a decay constant λ (his value depend on the isotope). The coefficients matrix is:

In[5]:=
```
A =
 CompartMatrix[3, {{1, 2, k12}, {1, 0, 1.9404}, {2, 3, k23} , {3, 0, 0.01155} ,
     {3, 1, 0.0462}}, λ] // Chop // TraditionalForm
```

Out[5]//TraditionalForm=
$$\begin{pmatrix} -\text{k12} - \lambda - 1.9404 & 0 & 0.0462 \\ \text{k12} & -\text{k23} - \lambda & 0 \\ 0 & \text{k23} & -\lambda - 0.05775 \end{pmatrix}$$

In[6]:=
```
ShowODE[{{-1.9404 - k12 - λ, 0, 0.0462}, {k12, -k23 - λ, 0}, {0, k23, -0.05775 - λ}},
  {0, 0, 0}, B, t, x] // TraditionalForm
```

Out[6]//TraditionalForm=
$$\{x_1'(t) = (-\lambda - \text{k12} - 1.9404)\, x_1(t) + 0.0462\, x_3(t) - 27.13\, e^{-24.08\, t} + 27.13\, e^{-2.86\, t} - 0.02\, e^{-0.147\, t} + 0.0194\, e^{-0.093\, t},$$
$$x_2'(t) = \text{k12}\, x_1(t) + (-\lambda - \text{k23})\, x_2(t),\ x_3'(t) = \text{k23}\, x_2(t) + (-\lambda - 0.05775)\, x_3(t),\ x_1(0) = 0,\ x_2(0) = 0,\ x_3(0) = 0\}$$

We represent the evolution of the iodine content in the compartment 1 where the samples will be taken (using $k_{12} = 0.8$ and $k_{23} = 0.0078$). We will refer to iodine 131 which has a radioactive half-life of 8 days, this meaning that radioactive decay constant $\lambda = \ln 2/8.02$ day^{-1}. Then the compartmental matrix is:

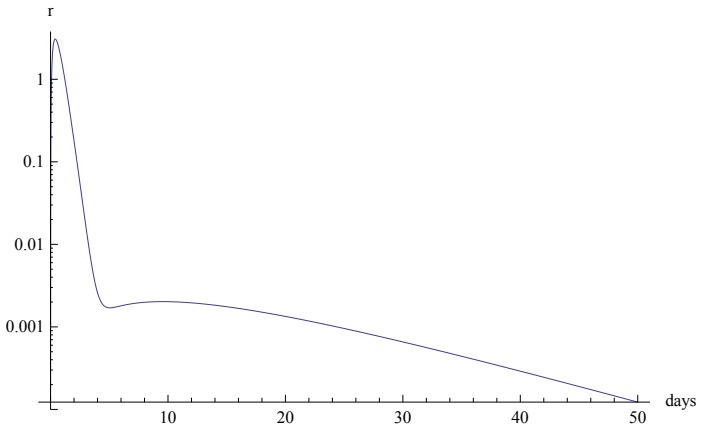
In[7]:=
```
iodine131matrix =
 CompartMatrix[3, {{1, 2, 0.8}, {1, 0, 1.9404}, {2, 3, 0.0078} ,
   {3, 0, 0.01155} , {3, 1, 0.0462}}, Log[2] / 8.02]
```

Out[7]=
```
{{-2.82683, 0., 0.0462}, {0.8, -0.0942273, 0.}, {0., 0.0078, -0.144177}}
```

In[8]:=
```
{x1[t_], x2[t_], x3[t_]} =
  {x1[t], x2[t], x3[t]} /. SystemDSolve[iodine131matrix, {0, 0, 0}, B, t, t, x] //
   Chop;
```

In[9]:=
```
LogPlot[x1[t], {t, 0, 50}, PlotRange → All, AxesLabel → {"days", "r"}]
```

Out[9]=



The form of the first interval it is explained because in during this interval (about 3 days) is happening a input b1(t) to compartment 1 and negligible flow from other compartments reach the compartment 1. In the second interval the opposite happens.

We wish obtain $x_{i(\beta)} = \{\partial x_i/\partial k_{12}, \partial x_i/\partial k_{23}\}$, $i = \{1,2,3\}$ to be used later for computing the Optimal Design.

We are going to apply different methods to obtain $x_{(\beta)}$ in order to make a comparison. Each method is computated in a new Mathematica session in order to compare the computation time.

In[1]:=
```
Quit[]
```

Method 1

Here will be obtain $x_{(\beta)}$ using eq(5) that we call the method of the extended matrix.

We call : $\text{Xa} = \left\{\text{x1a} = \frac{\partial x_1(t,\, k_{12})}{\partial k_{12}},\ \text{x2a} = \frac{\partial x_2(t,\, k_{12})}{\partial k_{12}},\ \text{x3a} = \frac{\partial x_3(t,\, k_{12})}{\partial k_{12}}\right\}$ and $\text{Ak12} = \mathbf{A}_{(k_{12})}$;

$\text{Xb} = \left\{\text{x1b} = \frac{\partial x_1(t,\, k_{23})}{\partial k_{23}},\ \text{x2a} = \frac{\partial x_2(t,\, k_{23})}{\partial k_{23}},\ \text{x3a} = \frac{\partial x_3(t,\, k_{23})}{\partial k_{23}}\right\}$ and $\text{Ak23} = \mathbf{A}_{(k_{23})}$

In[1]:=
```
Needs["Biokmod`SysModel`"]
```

```
  SysModel, version 1.5.1   2013-11-12
```

In[2]:=
```
A =
 CompartMatrix[3, {{1, 2, k12}, {1, 0, 1.9404}, {2, 3, k23} , {3, 0, 0.01155} ,
    {3, 1, 0.0462}}, Log[2] / 8.02] // Chop
```

Out[2]=
```
{{-2.02683 - k12, 0, 0.0462}, {k12, -0.0864273 - k23, 0}, {0, k23, -0.144177}}
```

In[3]:=
```
B = {-27.13 e^-24.08 t + 27.13 e^-2.86 t - 0.020 e^-0.147 t + 0.0194 e^-0.093 t, 0, 0};
```

In[4]:=
```
X = {x1[t], x2[t], x3[t]};
```

In[5]:=
```
eq1 = Thread[D[X, t] == A.X + B]
```

Out[5]=
```
{x1'[t] == -27.13 e^-24.08 t + 27.13 e^-2.86 t -
    0.02 e^-0.147 t + 0.0194 e^-0.093 t + (-2.02683 - k12) x1[t] + 0.0462 x3[t],
 x2'[t] == k12 x1[t] + (-0.0864273 - k23) x2[t], x3'[t] == k23 x2[t] - 0.144177 x3[t]}
```

In[6]:=
```
ic1 = {x1[0] == 0, x2[0] == 0, x3[0] == 0 };
```

In[7]:=
```
Xa = {x1a[t], x2a[t], x3a[t]};
```

In[8]:=
```
Xb = {x1b[t], x2b[t], x3b[t]};
```

```
In[9]:= Ak12 = D[A, k12]
Out[9]= {{-1, 0, 0}, {1, 0, 0}, {0, 0, 0}}

In[10]:= Ak23 = D[A, k23]
Out[10]= {{0, 0, 0}, {0, -1, 0}, {0, 1, 0}}

In[11]:= eq2 = Thread[D[Xa, t] == Ak12.X + A.Xa]
Out[11]= {x1a'[t] == -x1[t] + (-2.02683 - k12) x1a[t] + 0.0462 x3a[t],
          x2a'[t] == x1[t] + k12 x1a[t] + (-0.0864273 - k23) x2a[t],
          x3a'[t] == k23 x2a[t] - 0.144177 x3a[t]}

In[12]:= eq3 = Thread[D[Xb, t] == Ak23.X + A.Xb];

In[13]:= ic2 = {x1a[0] == 0, x2a[0] == 0, x3a[0] == 0 };

In[14]:= ic3 = {x1b[0] == 0, x2b[0] == 0, x3b[0] == 0 };
```

Now eq1 and eq2, and the respective initial conditions ic1 and ic2 are combined obtaining the eq4

```
In[15]:= eq4 = Join[eq1, eq2 , ic1, ic2] // Chop
Out[15]= {x1'[t] == -27.13 e^(-24.08 t) + 27.13 e^(-2.86 t) -
             0.02 e^(-0.147 t) + 0.0194 e^(-0.093 t) + (-2.02683 - k12) x1[t] + 0.0462 x3[t],
          x2'[t] == k12 x1[t] + (-0.0864273 - k23) x2[t], x3'[t] == k23 x2[t] - 0.144177 x3[t],
          x1a'[t] == -x1[t] + (-2.02683 - k12) x1a[t] + 0.0462 x3a[t],
          x2a'[t] == x1[t] + k12 x1a[t] + (-0.0864273 - k23) x2a[t],
          x3a'[t] == k23 x2a[t] - 0.144177 x3a[t], x1[0] == 0,
          x2[0] == 0, x3[0] == 0, x1a[0] == 0, x2a[0] == 0, x3a[0] == 0}

In[16]:= eq5 = Join[eq1, eq3 , ic1, ic3] // Chop
Out[16]= {x1'[t] == -27.13 e^(-24.08 t) + 27.13 e^(-2.86 t) -
             0.02 e^(-0.147 t) + 0.0194 e^(-0.093 t) + (-2.02683 - k12) x1[t] + 0.0462 x3[t],
          x2'[t] == k12 x1[t] + (-0.0864273 - k23) x2[t], x3'[t] == k23 x2[t] - 0.144177 x3[t],
          x1b'[t] == (-2.02683 - k12) x1b[t] + 0.0462 x3b[t],
          x2b'[t] == k12 x1b[t] - x2[t] + (-0.0864273 - k23) x2b[t],
          x3b'[t] == x2[t] + k23 x2b[t] - 0.144177 x3b[t], x1[0] == 0,
          x2[0] == 0, x3[0] == 0, x1b[0] == 0, x2b[0] == 0, x3b[0] == 0}
```

Them eq4 and eq5 can be solved when specific values of t, k_{12} and k_{23} are given

```
In[17]:= fa[a_?NumberQ, b_?NumberQ, t1_?NumberQ] :=
          Evaluate[{x1a[t], x2a[t], x3a[t]} /.
             NDSolve[Evaluate[eq4 /. {k12 → a, k23 → b}], Join[X, Xa], {t, 0, 100}]] /. t → t1

In[18]:= fb[a_?NumberQ, b_?NumberQ, t1_?NumberQ] :=
          Evaluate[{x1b[t], x2b[t], x3b[t]} /.
             NDSolve[Evaluate[eq5 /. {k12 → a, k23 → b}], Join[X, Xb], {t, 0, 100}]] /. t → t1
```

Then $\boldsymbol{x}_{(p)}(t)$:{x1a[t] x1b[t]}.

```
In[19]:= X1[a_, b_, ti_] := { fa[a, b, ti][[1, 1]], fb[a, b, ti][[1, 1]]}
```

So for a=k12 = 0.80; b= k23= 0.0078, then $\boldsymbol{x}_{(p)}(t)$:{x1a[t], x2a[t],x3a[t]} for t = {1,10,30} are (The computation time, in s, is the first value of the Output.)

```
In[20]:= Map[X1[0.80, 0.0078, #] &, {1, 10, 30}] // AbsoluteTiming

Out[20]= {0.062506,
  {{-0.722177, 0.00668556}, {0.000125787, 0.115799}, {0.0000414039, 0.0350599}}}
```

Other option faster than the previous is using the new function (Mathematica 9 or later is requiered) `ParametricNDSolve`.

```
In[21]:= eq4a = ParametricNDSolve[eq4, {x1, x2, x3, x1a, x2a, x3a}, {t, 0, 100}, {k12, k23}];

In[22]:= eq5a = ParametricNDSolve[eq5, {x1, x2, x3, x1b, x2b, x3b}, {t, 0, 100}, {k12, k23}];

In[23]:= fa[a_?NumberQ, b_?NumberQ, t_?NumberQ] := x1a[a, b][t] /. eq4a

In[24]:= fb[a_?NumberQ, b_?NumberQ, t_?NumberQ] := x1b[a, b][t] /. eq5a

In[25]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}
```

So for a=k12 = 0.80; b= k23= 0.0078, then $\boldsymbol{x}_{(p)}(t)$:{x1a[t], x2a[t],x3a[t]} for t = {1,10,30} are (The computation time, in s, is the first value of the Output.)

```
In[26]:= Map[X1[0.80, 0.0078, #] &, {1, 10, 30}] // AbsoluteTiming

Out[26]= {0., {{-0.722177, 0.00668556}, {0.000125787, 0.115799}, {0.0000414039, 0.0350599}}}

In[27]:= Quit[]
```

Method 2

This method give the solution as function of the unknown parameters (k12 and k23) then when are given specific values of k12 and k23 the ODE is numerically solved and the derivatives $x_{(\beta)}$ evaluated numerically in each point. To solve the ODE is used the Biomod function SystemDSolve (it applies the *Mathematica* function NDSolve).

```
In[1]:= Needs["Biokmod`SysModel`"]

  SysModel, version 1.5.1   2013-11-12

In[2]:= A[k12_, k23_] =
     CompartMatrix[3, {{1, 2, k12}, {1, 0, 1.9404}, {2, 3, k23} , {3, 0, 0.01155} ,
        {3, 1, 0.0462}}, Log[2] / 8.02] // Chop

Out[2]= {{-2.02683 - k12, 0, 0.0462}, {k12, -0.0864273 - k23, 0}, {0, k23, -0.144177}}

In[3]:= model[t1_?NumberQ, k12_?NumberQ, k23_?NumberQ] :=
       model[t1, k12, k23] = {x1[t1], x2[t1], x3[t1]} /.
          SystemNDSolve[A[k12, k23], {0, 0, 0},
           {-27.13 e^-24.08 t + 27.13 e^-2.86 t - 0.020 e^-0.147 t + 0.0194 e^-0.093 t, 0, 0} ,
           {t, 0, 100}, t1, x];
```

Note: Instead of the before function can be used the following function that solved analytically the ODE when k12 and k23 take numeric values but the take of computation is too longer

```
model[t1_?NumberQ, k12_?NumberQ, k23_?NumberQ] :=
  model[t1, k12, k23] = {x1[t1], x2[t1], x3[t1]} /. SystemDSolve[{ ...} , t, t1, x];
```

We use the package NumericalCalculus to compute the numerical derivations

```
In[4]:= Needs["NumericalCalculus`"]
```

We call fa(t1, k_{12}, k_{23})=$\left\{\frac{\partial x_1(t,k_{12})}{\partial k_{12}}, \frac{\partial x_2(t,k_{12})}{\partial k_{12}}, \frac{\partial x_3(t,k_{12})}{\partial k_{12}}\right\}$ and fb(t1, k_{12}, k_{23})=$\left\{\frac{\partial x_1(t,k_{23})}{\partial k_{23}}, \frac{\partial x_2(t,k_{23})}{\partial k_{23}}, \frac{\partial x_3(t,k_{23})}{\partial k_{23}}\right\}$ (for convenience we write a1, b2 instead of k_{12}, k_{23})

Note that sometime the option ND[model[ti,x,b1],x,a1,Scale->.01] should be used (see ND Help)

```
In[5]:= fa[a1_, b1_, ti_] := ND[model[ti, x, b1], x, a1]

In[6]:= fb[a1_, b1_, ti_] := ND[model[ti, a1, y], y, b1]
```

Now we test the method using the same values that in the previous example, that is k12 = 0.80; k23 = 0.0078;

```
In[7]:= X1[a_, b_, ti_] := { fa[a, b, ti][[1]], fb[a, b, ti][[1]]}

In[8]:= Map[X1[0.80, 0.0078, #] &, {1, 10, 30}] // AbsoluteTiming

Out[8]= {0.406247,
         {{-0.722124, 0.00666625}, {0.000125691, 0.115797}, {0.0000413294, 0.0349157}}}
```

The solution is almost the same that the obtained using Method 1 and the computation time is a bit bigger.

```
In[9]:= Quit[]
```

Method 3

This method is similar to Method 2 but here is used the new *Mathematica* (9 o later) funcion `ParametricNDSolve`.

```
In[1]:= Needs["Biokmod`SysModel`"]

  SysModel, version 1.5.1   2013-11-12

In[2]:= A =
         CompartMatrix[3, {{1, 2, k12}, {1, 0, 1.9404}, {2, 3, k23} , {3, 0, 0.01155} ,
            {3, 1, 0.0462}}, Log[2] / 8.02] // Chop

Out[2]= {{-2.02683 - k12, 0, 0.0462}, {k12, -0.0864273 - k23, 0}, {0, k23, -0.144177}}

In[3]:= B = {-27.13 e^(-24.08 t) + 27.13 e^(-2.86 t) - 0.020 e^(-0.147 t) + 0.0194 e^(-0.093 t), 0, 0};

In[4]:= eqs = ShowODE[A, {0, 0, 0}, B, t, x]

Out[4]= {x1'[t] == -27.13 e^(-24.08 t) + 27.13 e^(-2.86 t) - 0.02 e^(-0.147 t) + 0.0194 e^(-0.093 t) +
            (-2.02683 - k12) x1[t] + 0.0462 x3[t], x2'[t] == k12 x1[t] + (-0.0864273 - k23) x2[t],
         x3'[t] == k23 x2[t] - 0.144177 x3[t], x1[0] == 0, x2[0] == 0, x3[0] == 0}

In[5]:= sol = ParametricNDSolve[eqs, {x1, x2, x3}, {t, 0, 100}, {k12, k23}]

Out[5]= {x1 → ParametricFunction[<>],
         x2 → ParametricFunction[<>], x3 → ParametricFunction[<>]}

In[6]:= fa[a1_?NumberQ, b_?NumberQ, t_?NumberQ] := D[x1[a, b], a][t] /. a → a1 /. sol

In[7]:= fb[a_?NumberQ, b1_?NumberQ, t_?NumberQ] := D[x1[a, b], b][t] /. b → b1 /. sol

In[8]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}
```

So for a=k12 = 0.80; b= k23= 0.0078, then $x_{(p)}(t)$:{x1a[t], x2a[t],x3a[t]} for t = {1,10,30} are (The computation time, in s, is the first value of the Output.)

```
In[9]:= Map[X1[0.80, 0.0078, #] &, {1, 10, 30}] // AbsoluteTiming

Out[9]= {0.249968,
         {{-0.722177, 0.00668556}, {0.000125827, 0.115799}, {0.0000414037, 0.0350599}}}

In[10]:= Quit[]
```

The solution is almost the same that the obtained using Method 1 and 2 and the computation time is similar.

Method 4

In this case we will obtain again $\left\{\frac{\partial x_1(t,k_{12})}{\partial k_{12}}, \frac{\partial x_2(t,k_{12})}{\partial k_{12}}, \frac{\partial x_3(t,k_{12})}{\partial k_{12}}\right\}$ but in this case we will use eq(8) with $\mathbf{x}_0$=0

$\boldsymbol{x}_{(p)}(t) = \text{Exp}(\mathbf{A}\,t) * \mathbf{A}_{(p)}\,\text{Exp}(\mathbf{A}\,t) * \mathbf{b}(t)$

```
In[1]:= A = {{-2.026827329246876 - k12, 0, 0.0462}, {k12, -0.08642732924687598 - k23, 0},
          {0, k23, -0.14417732924687598}};

In[2]:= B = {-27.13 e^(-24.08 t) + 27.13 e^(-2.86 t) - 0.020 e^(-0.147 t) + 0.0194 e^(-0.093 t), 0, 0};

In[3]:= func[t1_?NumberQ, a_?NumberQ, b_?NumberQ, k_] := Module[{m1, AExp},
        m1 = MatrixExp[A t] /. {k12 → a, k23 → b} // ExpandAll // Chop;
        AExp = Map[Integrate[#1, {tau, 0, t}] &,
             Evaluate[m1 /. t -> t - tau].Evaluate[D[A, k].Evaluate[m1 /. t -> tau]],
             {1}];
        Map[Integrate[#1, {tau, 0, t}] &,
             Evaluate[AExp /. t -> t - tau].Evaluate[B /. t -> tau], {1}] /. t → t1]
```

The solution of fa is wrong (see the solution obtained with Method 1 and 2) and the time of computation is too long. We discart this method

```
In[4]:= {func[30, 0.80, 0.0078, k12], func[30, 0.80, 0.0078, k23]} // AbsoluteTiming

Out[4]= {86.933725,
          {{-0.00159739, -0.415486, 0.0167}, {0.0350599, -3.978464351720, 2.082955628269}}}
```

Note that this function is equivalente to the previous function: `X1[a_,b_,ti_]:={fa[a,b,ti][[1]], fb[a,b,ti][[1]]}`

Conclusion: Method 1 and 3 are very fast and they are also the easiest for programming. We will compare both methods in a OED

```
In[5]:= Quit[]
```

Optimal experiment design

We will suppose that k_{12} and k_{23} are unknown, although we know that their values will be about $k_{12} = 0.8$ and $k_{23} = 0.0078$. We wish estimate them taken experiment data from compartment 1 . The problem consist on decide by DOE the best moment to taken the sample. We will use *D*-optimal design.

Method 1

Here we will the optimal design experiment computing the derivatives using the method 1 that we have yet described

We wish find $t:\{t_0,..,\ t_i,\ ...\ t_n\}$ of the model given by eq. (4). (or (5)) using *D*-optimal design when the analitycal expression of $x_1(t,\ a,\ b)$ can not be found. [$f(t,\beta) = x_1(t,\ k_{12},\ k_{23})$]

1.- It is defined a model $f(t,\ \beta)$ where the unknown parameters are $\beta = \{a,\ b\}$. In our case we call $\beta = \{k_{12},\ k_{23}\}$. [We write eq4a and eq5a obtained when the method 1 has beed described]

```
In[1]:= eq4a = ParametricNDSolve[
          {x1'[t] == -27.13 e^(-24.08 t) + 27.13 e^(-2.86 t) - 0.02 e^(-0.147 t) + 0.0194 e^(-0.093 t) +
             (-2.026827 - k12) x1[t] + 0.0462 x3[t],
           x2'[t] == k12 x1[t] + (-0.086427 - k23) x2[t], x3'[t] == k23 x2[t] - 0.1441773 x3[t],
           x1a'[t] == -x1[t] + (-2.0268 - k12) x1a[t] + 0.0462` x3a[t],
           x2a'[t] == x1[t] + k12 x1a[t] + (-0.086427 - k23) x2a[t],
           x3a'[t] == k23 x2a[t] - 0.144177` x3a[t], x1[0] == 0, x2[0] == 0, x3[0] == 0,
           x1a[0] == 0, x2a[0] == 0, x3a[0] == 0}, {x1, x2, x3, x1a, x2a, x3a},
          {t, 0, 100}, {k12, k23}];
```

```
In[2]:= eq5a = ParametricNDSolve[
          {x1'[t] == -27.13 e^(-24.08` t) + 27.13 e^(-2.86 t) - 0.02 e^(-0.147 t) + 0.0194 e^(-0.093 t) +
              (-2.0268 - k12) x1[t] + 0.0462 x3[t], x2'[t] == k12 x1[t] + (-0.0864 - k23) x2[t],
           x3'[t] == k23 x2[t] - 0.1442 x3[t], x1b'[t] == (-2.0268 - k12) x1b[t] + 0.0462` x3b[t],
           x2b'[t] == k12 x1b[t] - x2[t] + (-0.0864 - k23) x2b[t],
           x3b'[t] == x2[t] + k23 x2b[t] - 0.14418 x3b[t], x1[0] == 0, x2[0] == 0,
           x3[0] == 0, x1b[0] == 0, x2b[0] == 0, x3b[0] == 0}, {x1, x2, x3, x1b, x2b, x3b},
          {t, 0, 100}, {k12, k23}];
```

2.-Now it is computed $\nabla(f(t),\ \{a,\ b\}) = \{\frac{\mathrm{df}(t)}{\mathrm{da}}, \frac{\mathrm{df}(t)}{\mathrm{db}}\}$,

```
In[3]:= fa[a_?NumberQ, b_?NumberQ, t_?NumberQ] := x1a[a, b][t] /. eq4a
```

```
In[4]:= fb[a_?NumberQ, b_?NumberQ, t_?NumberQ] := x1b[a, b][t] /. eq5a
```

3.- We need to define the number of points n to be used in the optimal design.

4.- It is evaluated ∇(f(t), β) at points t:{t_0,..., t_n}, obtaining $X = \{X_1,\ ...,\ X_p\}$ with $X_1 = \{\frac{\mathrm{df}(t_0)}{\mathrm{d}\beta_1},\ ...,\ \frac{\mathrm{df}(t_n)}{\mathrm{d}\beta_1}\}$, ..., $X_p = \{\frac{\mathrm{df}(t_0)}{\mathrm{d}\beta_p},\ ...,\ \frac{\mathrm{df}(t_n)}{\mathrm{d}\beta_p}\}$.

Because the sample will be taken in compartment 1 , we extract of fa and fb the derivatives corresponding to $x_1(t)$

```
In[5]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}
```

5.- A typical election for compute the covariance matrix is assumed that that the relationship between samples decays exponentially with increasing time-distance between them, that is $\Gamma = \{l_{ij}\}$ with l_{ij}= exp $\{\rho|t_j - t_j|\}$.For computational purpose we have found more appropriate to use the distance $d_i = t_i - t_{i-1}$, instead of t_i, then $t_i = \sum_i d_i$ being $d_0 = t_0$. That is for a two points design . We suppose a 3-points design. The first is defined by the user

Γ where

```
In[6]:= Γ = {{1, e^(-ρ d1), e^(-ρ (d1+d2))}, {e^(-ρ d1), 1, e^(-ρ d2)}, {e^(-ρ (d1+d2)), e^(-ρ d2), 1}};
```

6.- Now it is computed the covariance matrix $\Sigma = \sigma^2\ \Gamma$

```
In[7]:= Σ = σ^2 * Γ;
```

We assume

```
In[8]:= ρ = 1; σ = 1;
```

We will also need give the initial values of β the standard deviation of the measures. We also assumed k12= 0.80, k23=0.0078

7.- Then we can obtain the information matrix

M = $X^T\ \Sigma^{-1}\ X$

m := X . Inverse[Σ]. Transpose[X];

```
In[9]:= m1[ti_] := Transpose[Map[X1[0.80, 0.0078, #] &, ti] ]. Inverse[Σ].
          Map[X1[0.80, 0.0078, #] &, ti]
```

8.- Finally the determinant of the information matrix is maximized as function of d0, d1 and d2. We constrain the d values to a maximun of t=50 becouse to longer time the concentration will be very low (lower than the detection limit)

```
In[10]:= obj[d0_?NumericQ, d1_?NumericQ, d2_?NumericQ] := Det[m1[{d0, d1 + d0, d0 + d1 + d2}]]
```

```
In[11]:= sol1 = NMaximize[ {obj[d0, d1, d2], 0 < d0 < 50, 0.02 < d1 < 50,  0.02 < d2 < 50},
            {d0, d1, d2}] // Timing
```

InterpolatingFunction::dmval :
Input value {110.013} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

InterpolatingFunction::dmval :
Input value {110.013} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

InterpolatingFunction::dmval :
Input value {110.013} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

General::stop : Further output of InterpolatingFunction::dmval will be suppressed during this calculation. ≫

```
Out[11]= {1.812500, {0.0160671, {d0 → 0.748667, d1 → 7.23841, d2 → 3.66134}}}
```

```
In[12]:= Quit[]
```

Method 3

Here we will the optimal design experiment computing the derivatives using the method 3 that we have yet described.

1.- It is defined a model $f(t,\ \beta)$ where the unknown parameters are $\beta = \{a,\ b\}$. In our case we call $\beta = \{k_{12},\ k_{23}\}$. [We write the ODE of the system obtained when we described the method 3

```
In[1]:= sol = ParametricNDSolve[
            {x1'[t] == -27.13` e^-24.08` t + 27.13` e^-2.86` t - 0.02` e^-0.147` t + 0.0194` e^-0.093` t +
                (-2.026827` - k12) x1[t] + 0.0462` x3[t],
             x2'[t] == k12 x1[t] + (-0.08643` - k23) x2[t], x3'[t] == k23 x2[t] - 0.1442` x3[t],
             x1[0] == 0, x2[0] == 0, x3[0] == 0}, {x1, x2, x3}, {t, 0, 100}, {k12, k23}];
```

2.-Now it is computed $\nabla(f(t),\ \{a,\ b\}) = \{\frac{\mathrm{df}(t)}{\mathrm{da}}, \frac{\mathrm{df}(t)}{\mathrm{db}}\}$,

```
In[2]:= fa[a1_?NumberQ, b_?NumberQ, t_?NumberQ] := D[x1[a, b], a][t] /. a → a1 /. sol
```

```
In[3]:= fb[a_?NumberQ, b1_?NumberQ, t_?NumberQ] := D[x1[a, b], b][t] /. b → b1 /. sol
```

3.- We need to define the number of points n to be used in the optimal design.

4.- It is evaluated ∇(f(t), β) at points t:{t_0,..., t_n}, obtaining $X = \{X_1,\ ...,\ X_p\}$ with $X_1 = \{\frac{\mathrm{df}(t_0)}{\mathrm{d}\beta_1},\ ...,\ \frac{\mathrm{df}(t_n)}{\mathrm{d}\beta_1}\}$, ..., $X_p = \{\frac{\mathrm{df}(t_0)}{\mathrm{d}\beta_p},\ ...,\ \frac{\mathrm{df}(t_n)}{\mathrm{d}\beta_p}\}$.
Because the sample will be taken in compartment 1 , we extract of fa and fb the derivatives corresponding to $x_1(t)$

```
In[4]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}
```

5.- A typical election for compute the covariance matrix is assumed that that the relationship between samples decays exponentially with increasing time-distance between them, that is $\Gamma = \{l_{ij}\}$ with l_{ij}= exp $\{\rho|t_j - t_j|\}$.For computational purpose we have found more appropriate to use the distance $d_i = t_i - t_{i-1}$, instead of t_i, then t_i= $\sum_i d_i$ being $d_0 = t_0$. That is for a two points design . We suppose a 3-points design. The first is defined by the user

Γ where

```
In[5]:= Γ = {{1, e^-ρ d1, e^-ρ (d1+d2)}, {e^-ρ d1, 1, e^-ρ d2}, {e^-ρ (d1+d2), e^-ρ d2, 1}};
```

6.- Now it is computed the covariance matrix $\Sigma = \sigma^2\ \Gamma$

```
In[6]:= Σ = σ^2 * Γ;
```

We assume

```
In[7]:= ρ = 1; σ = 1;
```

We will also need give the initial values of β the standard deviation of the measures. We also assumed k12= 0.80, k23=0.0078

7.- Then we can obtain the information matrix

M = $X^T \Sigma^{-1} X$

m := X . Inverse[Σ]. Transpose[X];

```
In[8]:= m1[ti_] := Transpose[Map[X1[0.80, 0.0078, #] &, ti] ]. Inverse[Σ].
          Map[X1[0.80, 0.0078, #] &, ti]
```

8.- Finally the determinant of the information matrix is maximized as function of d0, d1 and d2. We constrain the d values to a maximun of t=50 becouse to longer time the concentration will be very low (lower than the detection limit)

```
In[9]:= obj[d0_?NumericQ, d1_?NumericQ, d2_?NumericQ] := Det[m1[{d0, d1 + d0, d0 + d1 + d2}]]
```

```
In[10]:= sol1 = NMaximize[ {obj[d0, d1, d2], 0 < d0 < 50, 0.02 < d1 < 50, 0.02 < d2 < 50},
            {d0, d1, d2}] // Timing

InterpolatingFunction::dmval :
  Input value {110.013} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

InterpolatingFunction::dmval :
  Input value {110.013} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

InterpolatingFunction::dmval :
  Input value {110.013} lies outside the range of data in the interpolating function. Extrapolation will be used. ≫

General::stop : Further output of InterpolatingFunction::dmval will be suppressed during this calculation. ≫

Out[10]= {2.031250, {0.0160593, {d0 → 0.748664, d1 → 7.23683, d2 → 3.661}}}
```

Conclusion: The time of computation of Method 1 and 3 are practically the same, method 1 a bit faster tham method 3,, but Method 3 is the easiast for programming

Here it is shown graphically the iteration process of d1 and de

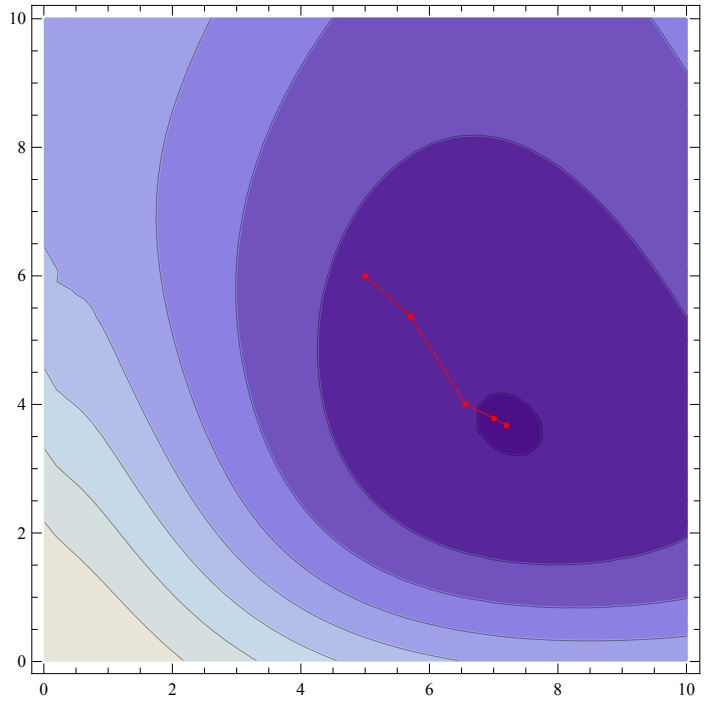
```
In[11]:= d0 = 0.748664;
```

```
In[12]:= FindMaximum[{obj[d0, d1, d2], 0.02 < d1 < 10, 0.02 < d2 < 10}, {{d1, 5}, {d2, 6}},
           StepMonitor :→ Print[{"d1:", d1, "d2:", d2}]]

{d1:, 5., d2:, 6.}

{d1:, 5.71153, d2:, 5.36494}

{d1:, 6.55693, d2:, 4.00819}

{d1:, 7.00631, d2:, 3.78098}

{d1:, 7.1993, d2:, 3.67541}

Out[12]= {0.0160593, {d1 → 7.23569, d2 → 3.66131}}
```

```
In[13]:= pts =
           Reap[FindMaximum[ {obj[d0, d1, d2], 0.02 < d1 < 10, 0.02 < d2 < 10},
              {{d1, 5}, {d2, 6}}, StepMonitor :→ Sow[{d1, d2}]]][[2, 1]];
```

In[14]:=
```
ContourPlot[obj[d0, d1, d2] // Minus, {d1, 0.02, 10}, {d2, 0.02, 10},
 Epilog → {Red, Line[pts], Point[pts]}]
```

Out[14]=



In[15]:=
```
Quit[]
```

Biokinetic model of Ciprofloxacin and Ofloxacin

The model

The following differential equations result from the model ofA. Sánchez-Navarro, C. Casquero, and M. Weiss, 'Distribution of Ciprofloxacin and Ofloxacin in the Isolated Hindlimb of the Rat', Pharmaceutical Research, 16: 587-591 (1999):

$$c_{out}'[t] + \left(\frac{Q}{V_p} + \frac{PS}{V_p}\right) c_{out}[t] - \frac{PS}{V_p} c_{Tu}[t] = \frac{Q}{V_p} c_{in}[t]$$

$$c_{Tu}'[t] + \left(\frac{PS}{V_{Tu}} + k_{on}\right) c_{Tu}[t] - \frac{PS}{V_{Tu}} c_{out}[t] - k_{off} \frac{V_{Tb}}{V_{Tu}} C_{Tb}[t] = 0$$

$$c_{Tb}'[t] + k_{off}\, c_{Tb}[t] - k_{on} \frac{V_{Tu}}{V_{Tb}} c_{Tu}[t] = 0$$

Initial conditions: $c_{out}[0] = 0,\ c_{Tu}[0] = 0,\ c_{Tb}[0] = 0$

$c_i(t)$, with i = {out, Tu, Tb}, represents the concentration of a sustance in differentes part of the models. We consider the case where with Q = 3 mL min^{-1}, V_{Tu} = 6.411, V_p, 0.973, V_{Tb} = 1, and PS = 2.714, then replacing V_{Tu}, V_p, V_{Tb}, PS, Q for their values where for convenients we call $x_1(t)$, $x_2(t)$ and $x_3(t)$ intead of $c_{out}(t)$, $c_{Tu}(t)$ and $c_{Tb}(t)$

$$\begin{aligned} x_1'(t) &= -5.87256\, x_1(t) + 2.7893\, x_2(t) + 3.08325\, c_{int}(t) \\ x_2'(t) &= 0.423335\, x_1(t) + (-k_{on} - 0.423335)\, x_2(t) + 0.15598\, k_{off}\, x_3(t) \\ x_3'(t) &= 6.411\, k_{on}\, x_2(t) - k_{off}\, x_3(t) \\ x_1(0) &= x_2(0) = x_3(0) = 0 \end{aligned} \tag{9}$$

We use $c_{in}(t) = 13\,610.1\, t\, e^{-11.216\, t}$ (According to G. Sanchez Biokmod: A Mathematica toolbox for modeling Biokinetic Systems". *Mathematica in Education and Research*: 10 (2) 2005. ISSN/ISBN: 1096-3324 using the experimental data obtained by of A. Sánchez-Navarro et al.)

Then, on notation matrix:

$$\boldsymbol{x}(t) = \mathbf{x}_0\, \mathrm{Exp}(\mathbf{A}\, t) + \mathrm{Exp}(\mathbf{A}\, t) * \mathbf{b}(t) \tag{10}$$

where

$x(t) = \{x_1(t),\ x_2(t),\ x_3(t)\}^T$

Optimal experiment design

This model (eq9) is represented in *Mathematica* by the ODE

```
In[1]:= eqox = { x1'[t] == -5.87256 x1[t] + 2.78931 x2[t] + 41 963.3 t Exp[-11.216 t],
     x2'[t] == 0.423335 x1[t] + (-kon - 0.423335) x2[t] + 0.155982 koff x3[t],
   x3'[t] == 6.411 kon x2[t] - koff x3[t],
   x1[0] == 0, x2[0] == 0, x3[0] == 0};
```

We want to estimate the values of k_{on} and k_{off} making an experiment that consist on in measure the concentration of a compound at $\hat{x}_1(t)$ for $t:\{t_0,..,\ t_i,\ ...\ t_n\}$ an then estimate k_{on} and k_{off} by fitting $x_1(t,\ k_{on},\ k_{off})$.

The problem is can not be found an analitycal expresion of $x_1(t,\ k_{on},\ k_{off})$ because the ODE system eq. (4). (or (5)) can not be solved if k_{on}, k_{off} are parameters. However $x_1(t,\ k_{on},\ k_{off})$ has solution when k_{on}, k_{off} take numeric values, this fact is used by some nonlinear regresion method to estimated k_{on}, k_{off}.

We wish find $t:\{t_0,..,\ t_i,\ ...\ t_n\}$ of the model given by eq. (4). (or (5)) using *D*-optimal design when the analitycal expresion of $x_1(t,\ k_{on},\ k_{off})$ can not be found. [$f(t,\beta) = x_1(t,\ k_{on},\ k_{off})$]

1.- It is defined a model $f(t,\ \beta)$ where the unkown parameters are β = {kon, koff}.

```
In[2]:= sol = ParametricNDSolve[eqox , {x1, x2, x3}, {t, 0, 100} , {kon, koff}]

Out[2]= {x1 → ParametricFunction[<>],
  x2 → ParametricFunction[<>], x3 → ParametricFunction[<>]}
```

2.-Now it is computed $\nabla(f(t),\ \{a,\ b\}) = \{\frac{df(t)}{da}, \frac{df(t)}{db}\}$,

```
In[3]:= fa[a1_?NumberQ, b_?NumberQ, t_?NumberQ] := D[x1[a, b], a][t] /. a → a1 /. sol

In[4]:= fb[a_?NumberQ, b1_?NumberQ, t_?NumberQ] := D[x2[a, b], b][t] /. b → b1 /. sol
```

3.- Here is defined the number of points n to be used in the optimal design.

4.- It is evaluated $\nabla(f(t),\beta)$ at points $t:\{t_0,...,t_n\}$, obtaining $X = \{X_1,\ ...,\ X_p\}$ with $X_1 = \{\frac{df(t_0)}{d\beta_1},\ ...,\ \frac{df(t_n)}{d\beta_1}\}$, ..., $X_p = \{\frac{df(t_0)}{d\beta_p},\ ...,\ \frac{df(t_n)}{d\beta_p}\}$

```
In[5]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}

In[6]:= X1[0.7, 0.11, 0.5] // AbsoluteTiming

Out[6]= {0.093774, {-0.837181, 0.361595}}
```

Test OK(The same value that using Method 1)

6.- A typical election for compute the covariance matrix is assumed that that the relationship between samples decays exponentially with increasing time-distance between them, that is $\Gamma = \{l_{ij}\}$ with $l_{ij} = \exp\{\rho|t_j - t_j|\}$.For computational purpose we have found more appropriate to use the distance $d_i = t_i - t_{i-1}$, instead of t_i, then $t_i = \sum_i d_i$ being $d_0 = t_0$. That is for a two points design . We suppose a 3-points design. The first is defined by the user

Γ where

```
FoldList[Plus, Subscript[d, 0], Table[di, {i, n}]];
ff[i_, j_] := Which[i == j, 1, i < j, e^(-ρ Sum[dk, {k, i+1, j}]), i > j, e^(-ρ Sum[dk, {k, j+1, i}])];
nn = 2;
Γ = Array[ff, {nn + 1, nn + 1}]
```

6.- Now it is computed the convariance matrix $\Sigma = \sigma^2\ \Gamma$

We take:

```
In[7]:= ρ = 1; σ = 1;
```

We will also need give the initial values for the β parameters and the standard deviation of the measures. kon= 0.7, koff=0.11

7.- Then we can obtain the information matrix

M = $X^T \Sigma^{-1} X$

m := X . Inverse[Σ]. Transpose[X];

```
In[8]:= m1[ti_] := Transpose[Map[X1[0.7, 0.11, #] &, ti] ]. Inverse[Σ].
          Map[X1[0.7, 0.11, #] &, ti]
```

8.- Finally the determinant of the information matrix is maximized as function of d0, d1 and d2. We constrain the d values to a maximun of d_i=10 becouse to longer time the concentration will be very low (lower than the detection limit)

For n (number of observations)= 2

```
In[9]:= Γ = {{1, e^(-ρ d1)}, {e^(-ρ d1), 1}};

In[10]:= Σ = σ^2 * Γ;

In[11]:= m1[ti_] := Transpose[Map[X1[0.7, 0.11, #] &, ti] ]. Inverse[Σ].
           Map[X1[0.7, 0.11, #] &, ti]

In[12]:= obj[d0_?NumericQ, d1_?NumericQ] := Det[m1[{d0, d1 + d0}]]

In[13]:= sol2 = NMaximize[ {obj[d0, d1], 0.02 < d0 < 10, 0.02 < d1 < 10}, {d0, d1}]

Out[13]= {2426.54, {d0 → 1.57416, d1 → 4.4425}}
```

For n (number of observations)= 3

```
In[14]:= Γ = {{1, e^(-ρ d1), e^(-ρ (d1+d2))}, {e^(-ρ d1), 1, e^(-ρ d2)}, {e^(-ρ (d1+d2)), e^(-ρ d2), 1}};

In[15]:= Σ = σ^2 * Γ;

In[16]:= obj[d0_?NumericQ, d1_?NumericQ, d2_?NumericQ] := Det[m1[{d0, d1 + d0, d0 + d1 + d2}]]

In[17]:= sol3 = NMaximize[ {obj[d0, d1, d2], 0.02 < d0 < 10, 0.02 < d1 < 10, 0.02 < d2 < 10},
           {d0, d1, d2}]

Out[17]= {4322.28, {d0 → 1.56996, d1 → 3.58403, d2 → 3.15854}}
```

For n (number of observations)= 4

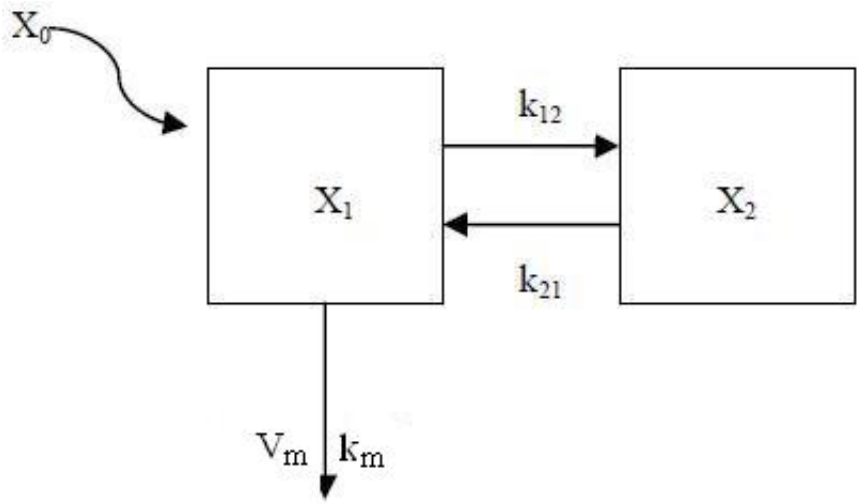
```
In[18]:= Γ = {{1, e^(-ρ d1), e^(-ρ (d1+d2)), e^(-ρ (d1+d2+d3))},
             {e^(-ρ d1), 1, e^(-ρ d2), e^(-ρ (d2+d3))},
             {e^(-ρ (d1+d2)), e^(-ρ d2), 1, e^(-ρ d3)},
             {e^(-ρ (d1+d2+d3)), e^(-ρ (d2+d3)), e^(-ρ d3), 1}};
           Σ = σ^2 * Γ;

In[19]:= m1[ti_] := Transpose[Map[X1[0.7, 0.11, #] &, ti] ]. Inverse[Σ].
           Map[X1[0.7, 0.11, #] &, ti]

In[20]:= obj[d0_?NumericQ, d1_?NumericQ, d2_?NumericQ, d3_?NumericQ] :=
          Det[m1[{d0, d1 + d0, d0 + d1 + d2, d0 + d1 + d2 + d3}]]

In[21]:= sol4 = NMaximize[ {obj[d0, d1, d2, d3], 0.02 < d0 < 10, 0.02 < d1 < 10,
             0.02 < d2 < 10, 0.02 < d3 < 10}, {d0, d1, d2, d3}]

Out[21]= {5871.95, {d0 → 1.55933, d1 → 2.90892, d2 → 2.59358, d3 → 3.07663}}
```

Conclusion: The observations will be taken: n, t0, t1,t2,t3}

```
In[22]:= {{"Observations 2:", d0, d1 + d0} /. sol2[[2]],
          {"Observations 3:", d0, d1 + d0, d0 + d1 + d2} /. sol3[[2]],
          {"Observations 4:", d0, d1 + d0, d0 + d1 + d2, d0 + d1 + d2 + d3} /. sol4[[2]]}

Out[22]= {{Observations 2:, 1.57416, 6.01666}, {Observations 3:, 1.56996, 5.15399, 8.31253},
          {Observations 4:, 1.55933, 4.46826, 7.06183, 10.1385}}

In[23]:= Quit[]
```

Michaelis-Menten

The model



In this case the drug transference between compartments will be considered as a linear kinetic process described by the transfer coefficients k12 and k21. However, the elimination process will be non-linear as it happens for instance in hepatic metabolism, and the elimination rate of the drug can mathematically be expressed by the Michaelis-Menten equation with parameters Vm=maximum transformation speed and km=Michaelis-Menten constant. The drug administration will be assumed to be an impulsive input (bolus).

1.- It is defined a model $f(t,\ \beta)$ where the unkown parameters are $\beta = \{a,\ b\}$.

Vmax = 0.2; km = 0.3

```
In[1]:= Vmax = 0.2; km = 0.3; V1 = 1;

In[2]:= eq1 = x1'[t] == -k12 x1[t] + k21 x2[t] - Vmax x1[t] / V1 / (km + x1[t] / V1);

In[3]:= eq2 = x2'[t] == k12 x1[t] - k21 x2[t];
```

It can be solved as function of k12 and k21:

```
In[4]:= sol = ParametricNDSolve[{eq1, eq2, x1[0] == 1, x2[0] == 0}, {x1, x2},
            {t, 0, 100}, {k12, k21}]

Out[4]= {x1 → ParametricFunction[<>], x2 → ParametricFunction[<>]}
```

We wish find $t:\{t_0,..,\ t_i,\ ...\ t_n\}$) using *D*-optimal design when the analitycal expresion of $x_1(t,\ k_a,\ k_b)$ can not be found. [f(t, β) $=x_1(t,\ k_a,\ k_b)$]

2.-Now it is computed $\nabla(f(t),\ \{a,\ b\}) = \{\frac{df(t)}{da}, \frac{df(t)}{db}\}$,

```
In[5]:= fa[a1_?NumberQ, b_?NumberQ, t_?NumberQ] := D[x1[a, b], a][t] /. a → a1 /. sol

In[6]:= fb[a_?NumberQ, b1_?NumberQ, t_?NumberQ] := D[x1[a, b], b][t] /. b → b1 /. sol
```

3.-It is defined the number of points n to be used in the optimal design.

4.- It is evaluated $\nabla(f(t), \beta)$ at points $t:\{t_0,..., t_n\}$, obtaining $X = \{X_1, ..., X_p\}$ with $X_1 = \{\frac{df(t_0)}{d\beta_1}, ..., \frac{df(t_n)}{d\beta_1}\}$, ..., $X_p = \{\frac{df(t_0)}{d\beta_p}, ..., \frac{df(t_n)}{d\beta_p}\}$

```
In[7]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}
```

We test using typical values of ki: k12 = 0.03, k21 = 0.02 for t:{1,3,10}

```
In[8]:= Map[X1[0.03, 0.02, #] &, {1, 3, 10}] // AbsoluteTiming
```

```
Out[8]= {0.062493, {{-0.867671, 0.0135637}, {-1.84457, 0.0962505}, {-0.248479, 0.167842}}}
```

6.- A typical election for compute the covariance matrix is assumed that that the relationship between samples decays exponentially with increasing time-distance between them, that is $\Gamma = \{l_{ij}\}$ with $l_{ij} = \exp\{\rho|t_j - t_j|\}$.For computational purpose we have found more appropriate to use the distance $d_i = t_i - t_{i-1}$, instead of t_i, then $t_i = \sum_i d_i$ being $d_0 = t_0$. That is for a two points design . We suppose a 3-points design. The first is defined by the user

Γ where

```
In[9]:= Γ = {{1, e^(-ρ d1), e^(-ρ (d1+d2))}, {e^(-ρ d1), 1, e^(-ρ d2)}, {e^(-ρ (d1+d2)), e^(-ρ d2), 1}};;
```

6.- Now it is computed the convariance matrix $\Sigma = \sigma^2\ \Gamma$

```
In[10]:= Σ = σ^2 * Γ;
```

We take:

```
In[11]:= ρ = 1; σ = 1;
```

```
In[12]:= Σ
```

```
Out[12]= {{1, e^-d1, e^-d1-d2}, {e^-d1, 1, e^-d2}, {e^-d1-d2, e^-d2, 1}}
```

We will also need give the initial values for the β parameters and the standard deviation of the measures. ka= 0.03, kb=0.02

7.- Then we can obtain the information matrix

M = $X^T\ \Sigma^{-1}\ X$

m := X . Inverse[Σ]. Transpose[X];

```
In[13]:= m1[ti_] := Transpose[Map[X1[0.03, 0.02, #] &, ti] ]. Inverse[Σ].
          Map[X1[0.03, 0.02, #] &, ti]
```

```
In[14]:= obj[d0_?NumericQ, d1_?NumericQ, d2_?NumericQ] := Det[m1[{d0, d1 + d0, d0 + d1 + d2}]]
```

```
In[15]:= sol1 = NMaximize[ {obj[d0, d1, d2], 0 < d0 < 10, 0 < d1 < 10, 0 < d2 < 10}, {d0, d1, d2}] //
          Timing
```

```
Out[15]= {3.421875, {0.189599, {d0 → 3.02083, d1 → 3.36409, d2 → 3.029}}}
```

```
In[16]:= {d0, d1 + d0, d0 + d1 + d2} /. sol1[[2, 2]]
```

```
Out[16]= {3.02083, 6.38492, 9.41392}
```

2.-Now it is computed $\nabla(f(t), \{a, b\}) = \{\frac{df(t)}{da}, \frac{df(t)}{db}\}$,

```
In[17]:= fa[a1_?NumberQ, b_?NumberQ, t_?NumberQ] := D[x1[a, b], a][t] /. a → a1 /. sol
```

```
In[18]:= fb[a_?NumberQ, b1_?NumberQ, t_?NumberQ] := D[x1[a, b], b][t] /. b → b1 /. sol
```

3.- Here is defined the number of points n to be used in the optimal design.

4.- It is evaluated $\nabla(f(t), \beta)$ at points $t:\{t_0,..., t_n\}$, obtaining $X = \{X_1, ..., X_p\}$ with $X_1 = \{\frac{df(t_0)}{d\beta_1}, ..., \frac{df(t_n)}{d\beta_1}\}$, ..., $X_p = \{\frac{df(t_0)}{d\beta_p}, ..., \frac{df(t_n)}{d\beta_p}\}$

```
In[19]:= X1[a_, b_, ti_] := { fa[a, b, ti], fb[a, b, ti]}
```

We test using typical values of ki: ka = 0.03, kb = 0.02

```
In[20]:= Map[X1[0.03, 0.02, #] &, {1, 3, 10}] // AbsoluteTiming
Out[20]= {0., {{-0.867671, 0.0135637}, {-1.84457, 0.0962505}, {-0.248479, 0.167842}}}
```

6.- A typical election for compute the covariance matrix is assumed that that the relationship between samples decays exponentially with increasing time-distance between them, that is $\Gamma = \{l_{ij}\}$ with $l_{ij}= \exp\{\rho|t_j - t_j|\}$.For computational purpose we have found more appropriate to use the distance $d_i = t_i - t_{i-1}$, instead of t_i, then $t_i= \sum_i d_i$ being $d_0 = t_0$. That is for a two points design . We suppose a 3-points design. The first is defined by the user

Γ where

```
In[21]:= Γ = {{1, e^(-ρ d1), e^(-ρ (d1+d2))}, {e^(-ρ d1), 1, e^(-ρ d2)}, {e^(-ρ (d1+d2)), e^(-ρ d2), 1}};;
```

6.- Now it is computed the convariance matrix $\Sigma = \sigma^2\ \Gamma$

```
In[22]:= Σ = σ^2 * Γ;
```

We take:

```
In[23]:= ρ = 1; σ = 1;
In[24]:= Σ
Out[24]= {{1, e^-d1, e^-d1-d2}, {e^-d1, 1, e^-d2}, {e^-d1-d2, e^-d2, 1}}
```

We will also need give the initial values for the β parameters and the standard deviation of the measures. ka= 0.03, kb=0.02

7.- Then we can obtain the information matrix

M = $X^T\ \Sigma^{-1}\ X$

m := X . Inverse[Σ]. Transpose[X];

```
In[25]:= m1[ti_] := Transpose[Map[X1[0.03, 0.02, #] &, ti] ]. Inverse[Σ].
          Map[X1[0.03, 0.02, #] &, ti]
In[26]:= sol1 = NMaximize[ {obj[d0, d1, d2], 0 < d0 < 10, 0 < d1 < 10, 0 < d2 < 10}, {d0, d1, d2}]
Out[26]= {0.189599, {d0 → 3.02083, d1 → 3.36409, d2 → 3.029}}
```

The observation should be taken (i):

```
In[27]:= {d0, d1 + d0, d0 + d1 + d2} /. sol1[[2]]
Out[27]= {3.02083, 6.38492, 9.41392}
```