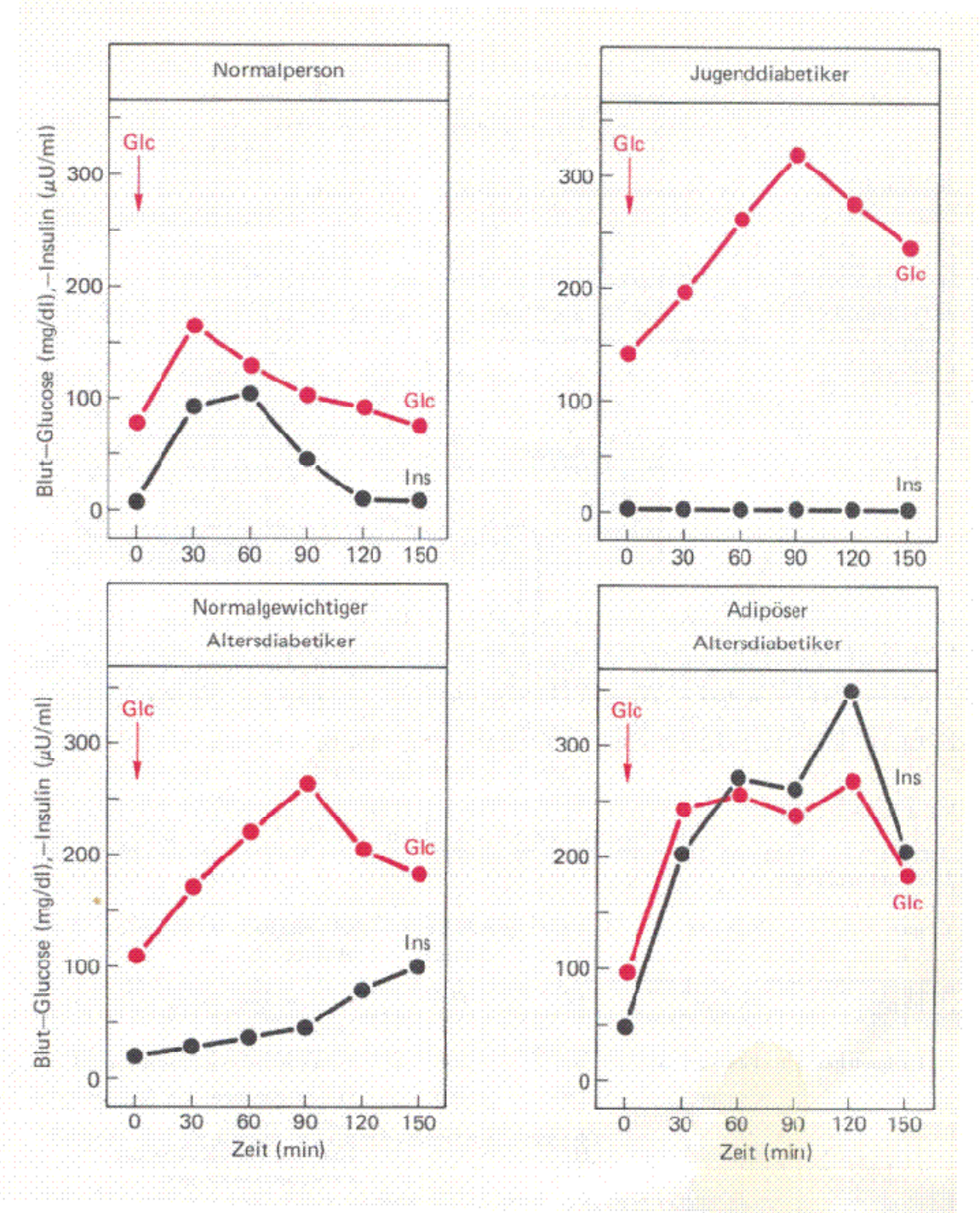
Analyzing the pharmakinetics model:

Reference:



Part 1 - ODE of the system: Solution of the OGTT equation:

In this section I use subscripted variables. First clear all existing values:

In[111]:=
```
ClearAll["Global`*"]
```

In[112]:=
```
Needs["Notation`"]
```

In[113]:=
```
vars = {g[t], i[t]};

eqns = {g'[t] == (If[t < τ, dose, dose * Exp[-0.05 * (t - τ)]] - m1 * g[t] - m2 * i[t]),
   i'[t] == m3 * g[t] - m4 * i[t]};

inits = {g[0] == 0, i[0] == 0};


τ = 15;
ρ = 0;
numberofparams = 4;
```

In[119]:=
```
conceqn = Simplify[DSolve[Join[eqns, inits], vars, t] ];
```

In[120]:=
```
m1v = 0.05; m2v = 0.05; m3v = 0.05; m4v = 0.05;  dosev = 10;
```

Part 2 - Plotting of the ODE:

(To Enter The Transpose T at data2 you have to enter ESC tr ESC)

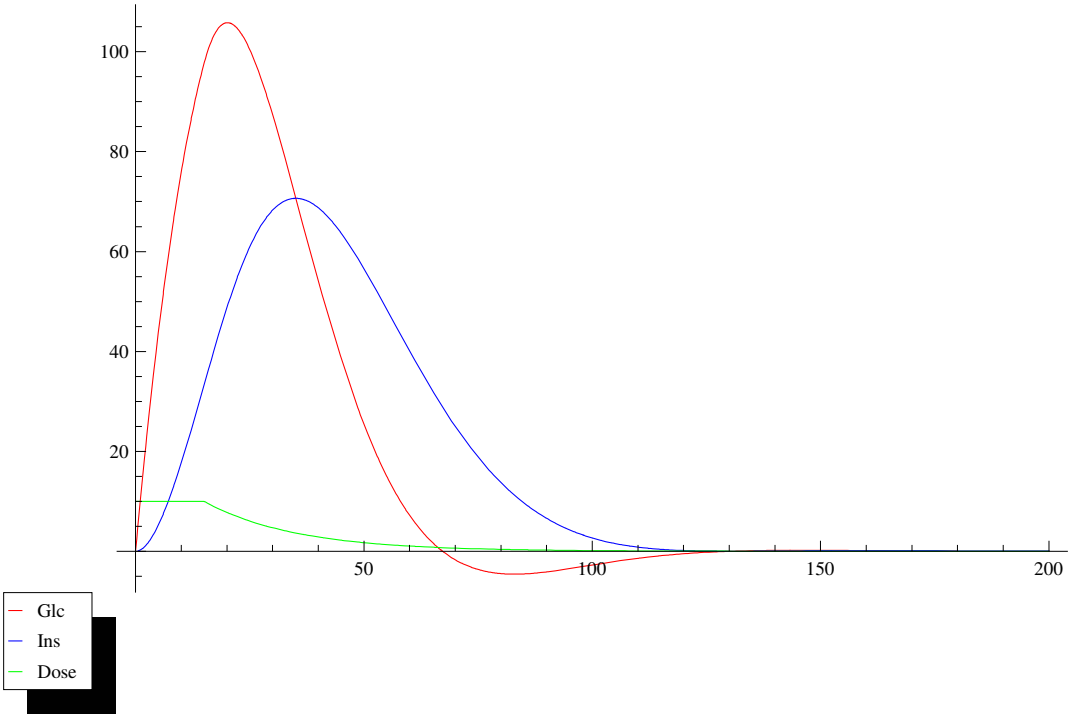
In[121]:=
```
Needs["PlotLegends`"]
```

In[122]:=
```
model = {vars[[1]] /. conceqn , vars[[2]] /. conceqn};
```

In[123]:=
```
p6 = Plot[{(model[[1]] /. {m1 → m1v, m2 → m2v, m3 → m3v, m4 → m4v, dose → dosev}),
    (model[[2]] /. {m1 → m1v, m2 → m2v, m3 → m3v, m4 → m4v, dose → dosev}),
    If[t < τ, dosev, dosev * Exp[-0.05 * (t - τ)]]}, {t, 0, 200},
  PlotRange → All, ImageSize → 500, PlotStyle → {Red, Blue, Green},
  PlotLegend → {"Glc", "Ins", "Dose"}, LegendSize → 0.2];
```

In[124]:=
```
Show[p6]
```

Out[124]=



Part 3 - Fitting the observed data:

The following data are the observed Glycose from Fig 1 a) and the treshold of yy1 is subtracted from the other yy of these data.
The data yy2 is the observed insulin

In[125]:=
```
xx = {0, 31.21, 61.53, 90.95, 126.19, 150};
yy = {77.32, 165.58, 129.69, 103.57, 91.64, 74.57};

treshold = yy[[1]]; yy = Map[ # - treshold &, yy]; data1 = {xx, yy}^T;
xx2 = xx;
yy2 = {0, 91.51, 104.24, 45.17, 8.11, 7.79};
data2 = {xx2, yy2}^T;
```

In[131]:=
```
p1 = ListPlot[{data1, data2}, PlotStyle → {Red, Blue} ];
f2a = (model[[1]] /. {m1 → m1op, m2 → m2op, m3 → m3op, m4 → m4op, dose → dosev});
 f2b = (model[[2]] /. {m1 → m1op, m2 → m2op, m3 → m3op, m4 → m4op, dose → dosev});
MyFIT = FindFit[data1, f2a, {{m1op, 0.035}, {m2op, 9.602 * 10^-6},
    {m3op, 1.08 * 10^-5}, {m4op, 0.0434}}, t, MaxIterations → 5000]
```

FindFit::sszero :
The step size in the search has become less than the tolerance prescribed by the PrecisionGoal option, but the gradient is larger than the tolerance specified by the AccuracyGoal option. There is a possibility that the method has stalled at a point that is not a local minimum. ≫

Out[134]= {m1op → 0.148511, m2op → -0.11366, m3op → 0.117403, m4op → 0.143462}

At this step I used the Sum-squared-errors which should be a minimum. To include both curves I have added the SSE of curve1 and SSE of curve2

In[135]:=
```
f3a := (model[[1]] /. { dose → dosev}); f3b := (model[[2]] /. { dose → dosev});
SSEa = Sum[((yy[[i]] - ((f3a /. {t → xx[[i]]})))^2), {i, Length[xx]}];
SSEb = Sum[((yy2[[i]] - ((f3b /. {t → xx2[[i]]})))^2), {i, Length[xx2]}];
```

In[136]:=
```
MyFITminOP =
 NMinimize[SSEa + SSEb, {{m1, 0, 0.1}, {m2, -0.04, 0.05}, {m3, 0, 0.5}, {m4, 0, 0.5}},
  MaxIterations → 50 000 , Method → {"RandomSearch", "PostProcess" → "KKT"} ]
```

LessEqual::nord : Invalid comparison with $0.270713 + 2.41089 \times 10^{-41}\, i$ attempted. ≫

LessEqual::nord : Invalid comparison with $0.270713 + 2.41089 \times 10^{-41}\, i$ attempted. ≫

LessEqual::nord : Invalid comparison with $0.364895 + 1.28107 \times 10^{-40}\, i$ attempted. ≫

General::stop : Further output of LessEqual::nord will be suppressed during this calculation. ≫

FindMinimum::fmns :
Starting value $\{0.270713 + 2.41089 \times 10^{-41}\, i, 0.364895 + 1.28107 \times 10^{-40}\, i, 0.310963 + 5.69138 \times 10^{-41}\, i, 0.26569 + 1.41663 \times 10^{-42}\, i\}$ contains numbers that are not real. ≫

FindMinimum::fmns :
Starting value $\{0.270713 + 2.41089 \times 10^{-41}\, i, 0.364895 + 1.28107 \times 10^{-40}\, i, 0.310963 + 5.69138 \times 10^{-41}\, i, 0.26569 + 1.41663 \times 10^{-42}\, i\}$ contains numbers that are not real. ≫

Out[136]= {632.079, {m1 → 0.0816374, m2 → -0.0191792, m3 → 0.0778494, m4 → 0.0572546}}

In[137]:=
```
ERR = Sqrt[MyFITminOP[[1]]]
```

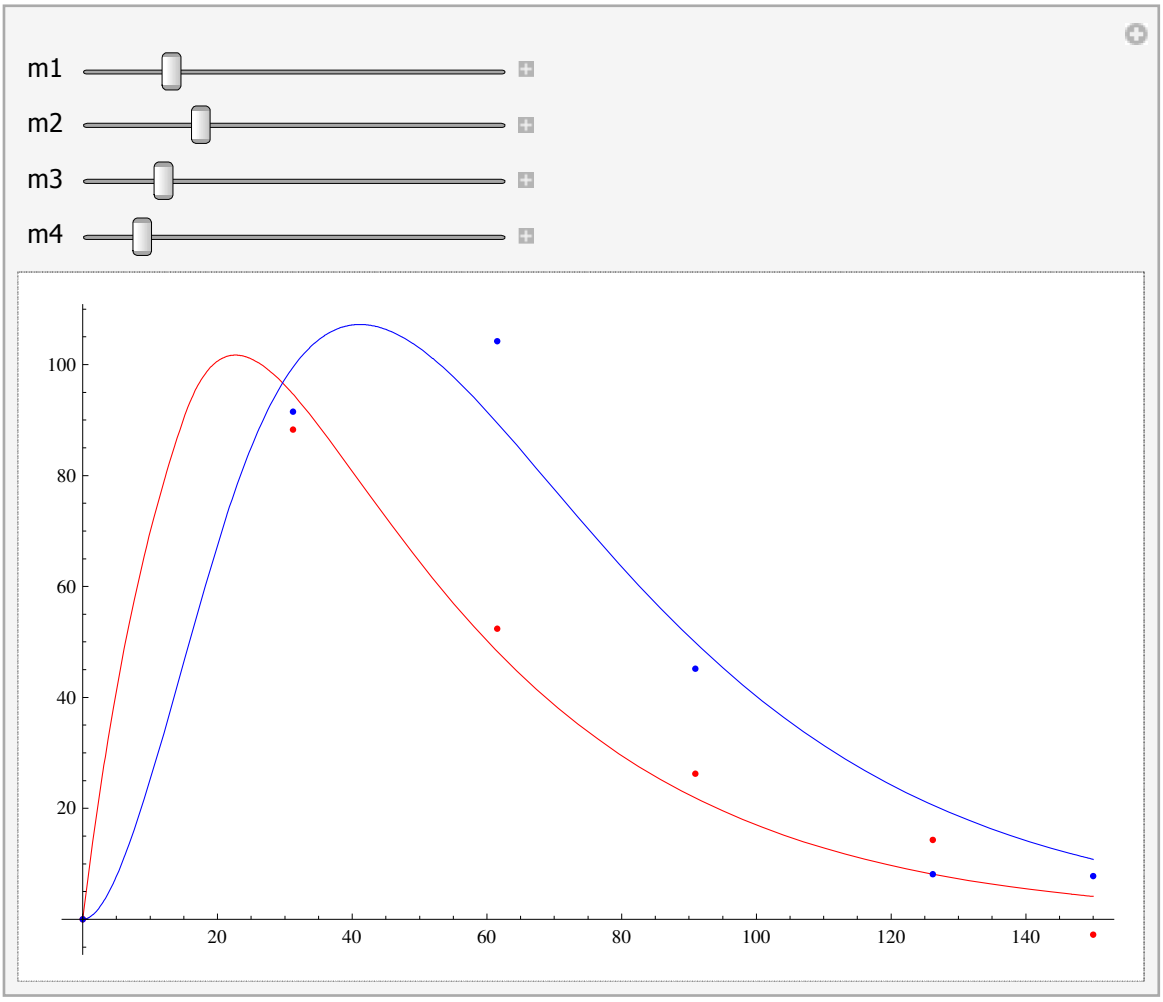
Out[137]= 25.1412

In[138]:=
```
MyFITmin = MyFITminOP[[2]] /. {m1 → m1op, m2 → m2op, m3 → m3op, m4 → m4op}
```

Out[138]=
```
{m1op → 0.0816374, m2op → -0.0191792, m3op → 0.0778494, m4op → 0.0572546}
```

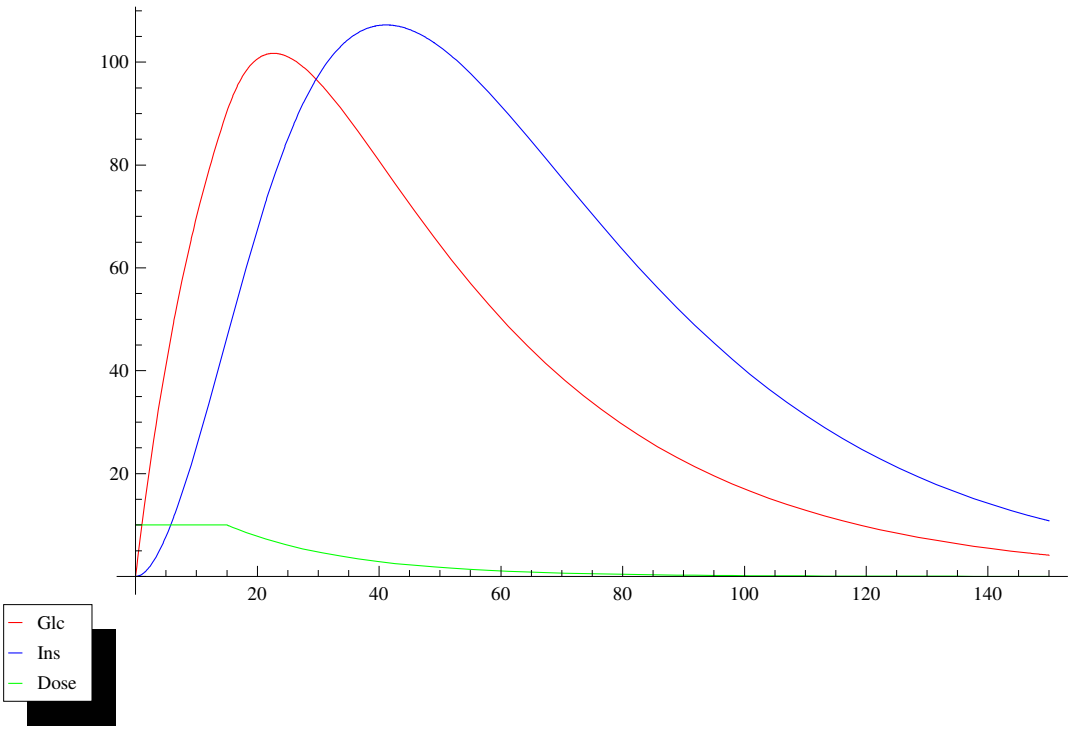
In[139]:=
```
Manipulate[
 p2 = Plot[{(f2a /. {m1op → m1, m2op → m2, m3op → m3, m4op → m4, doseop → dosev}) ,
    ((f2b /. {m1op → m1, m2op → m2, m3op → m3, m4op → m4, doseop → dosev}) * 1),
    If[t < τ, dose, dose * Exp[-0.05 * (t - τ)]]}, {t, 0, 150},
   PlotRange → All, ImageSize → 500, PlotStyle → {Red, Blue, Green}];
 Show[p2, p1],
 {{m1, Re[m1op] /. MyFITmin}, -0.01, 0.5},
 {{m2, Re[m2op] /. MyFITmin}, -0.2, 0.5},
 {{m3, Re[m3op] /. MyFITmin}, 0.00, 0.5},
 {{m4, Re[m4op] /. MyFITmin}, 0.005, 0.5}, SaveDefinitions → True

]
```

Out[139]=



In[140]:=
```
p2 = Plot[{(f2a  /. MyFITmin) , ((f2b /. MyFITmin)),
     If[t < τ, dosev, dosev * Exp[-0.05 * (t - τ)]] /. MyFITmin}, {t, 0, 150},
   PlotRange → All, ImageSize → 500, PlotStyle → {Red, Blue, Green},
   PlotLegend → {"Glc", "Ins", "Dose"}, LegendSize → 0.2];
Show[
 p2]
```

Out[141]=



Part 4 - Calculate the Standard Deviation with Fisher`s Info Matrix

In[142]:=
```
Symbolize[ m_ ]
Symbolize[ σ_ ]
```

In[144]:=
```
numDerivsTable[symDerivMat_, Xtime_, MyFIT_] :=
  Module[
   {numDerivM = Table[symDerivMat /. {t → Xtime[[i]], m1 → m1op, m2 → m2op, m3 → m3op,
         m4 → m4op, dose → dosev} /. MyFIT, {i, 1, Length[Xtime]}]},
   numDerivM];
DerivM = D[model[[1]], {{m1, m2, m3, m4}}];
MyFITminReal = {m1 → Re[m1op], m2 → Re[m2op], m3 → Re[m3op], m4 → Re[m4op]} /. MyFITmin /.
  {m1 → m1op, m2 → m2op, m3 → m3op, m4 → m4op, dose → dosev}
ND = Flatten[numDerivsTable[DerivM, xx, MyFITminReal], 1]; MatrixForm[ND]
```

Out[145]=
```
{m1op → 0.0816374, m2op → -0.0191792, m3op → 0.0778494, m4op → 0.0572546}
```

Out[146]//MatrixForm=

$$\begin{pmatrix} -4.30689\times10^{-13} & -1.72275\times10^{-12} & 0. & -4.30689\times10^{-13} \\ -1133.33 & -845.012 & 208.18 & -119.546 \\ -1082.17 & -1456.62 & 358.857 & -389.119 \\ -699.009 & -1198.29 & 295.215 & -433.868 \\ -354.024 & -699.774 & 172.399 & -308.647 \\ -212.476 & -444.702 & 109.558 & -212.869 \end{pmatrix}$$

In[147]:=
```
FisherInfoMat = Dot[NDᵀ, ND]; MatrixForm[FisherInfoMat]
```

Out[147]//MatrixForm=

$$\begin{pmatrix} 3.11463\times10^6 & 3.71384\times10^6 & -914952. & 1.01436\times10^6 \\ 3.71384\times10^6 & 4.95913\times10^6 & -1.22175\times10^6 & 1.49836\times10^6 \\ -914952. & -1.22175\times10^6 & 300993. & -369141. \\ 1.01436\times10^6 & 1.49836\times10^6 & -369141. & 494523. \end{pmatrix}$$

In[148]:=
```
FisherInfoMatRe = Map[Re[#] &, FisherInfoMat]; MatrixForm[FisherInfoMatRe]
```

Out[148]//MatrixForm=

$$\begin{pmatrix} 3.11463\times10^6 & 3.71384\times10^6 & -914952. & 1.01436\times10^6 \\ 3.71384\times10^6 & 4.95913\times10^6 & -1.22175\times10^6 & 1.49836\times10^6 \\ -914952. & -1.22175\times10^6 & 300993. & -369141. \\ 1.01436\times10^6 & 1.49836\times10^6 & -369141. & 494523. \end{pmatrix}$$

In[149]:=
```
degreeoffreedom = Length[xx] - numberofparams;
SSEan = Re[SSEa] /. (MyFITminReal /. {m1op → m1, m2op → m2, m3op → m3, m4op → m4})
```

Out[149]= {162.677}

In[150]:=
```
{√(SSEan / degreeoffreedom [[1]]) √(Diagonal[PseudoInverse[FisherInfoMat]])} //
 TraditionalForm
```

Out[150]//TraditionalForm=

$$(0.038231 \quad 0.0566069 \quad 0.0139459 \quad 0.107965)$$

In[151]:=
```
% // MatrixForm
```

Out[151]//MatrixForm=

$$\begin{pmatrix} 0.038231 & 0.0566069 & 0.0139459 & 0.107965 \end{pmatrix}$$

■ Results of the Standard-Deviations and system-parameters:

In[152]:=
```
{{σm1, σm2, σm3, σm4} = {Flatten[Map[ Re[#] &, %]]}
```

Out[152]= {{0.038231, 0.0566069, 0.0139459, 0.107965}}

In[153]:=
```
{{m1, m2, m3, m4} = {Map[Re[#] &, {m1op, m2op, m3op, m4op} /. MyFITmin]}
```

Out[153]= {{0.0816374, -0.0191792, 0.0778494, 0.0572546}}

■ To see that the INVERSE makes a warnings, here the output:

In[154]:=
```
Diagonal[Inverse[FisherInfoMatRe]]
```

Inverse::luc : Result for Inverse of badly conditioned matrix
{{3.11463×10^6, 3.71384×10^6, −914952., 1.01436×10^6}, {3.71384×10^6, ≪21≫, −≪22≫, 1.49836×10^6}, {≪1≫}, {1.01436×10^6, 1.49836×10^6, −369141., 494523.}}
may contain significant numerical errors. ≫

Out[154]= {0.0000179695, -1.05264×10^9, -1.73431×10^10, 0.000143307}

In[155]:=
```
PseudoInverse[FisherInfoMatRe] // TraditionalForm
```

Out[155]//TraditionalForm=

$$\begin{pmatrix} 0.0000179695 & -0.0000258808 & 6.37606\times10^{-6} & 0.0000463174 \\ -0.0000258808 & 0.0000393953 & -9.70554\times10^{-6} & -0.0000735231 \\ 6.37606\times10^{-6} & -9.70554\times10^{-6} & 2.39109\times10^{-6} & 0.0000181134 \\ 0.0000463174 & -0.0000735231 & 0.0000181134 & 0.000143307 \end{pmatrix}$$

In[156]:= **Diagonal[PseudoInverse[FisherInfoMatRe]]**

Out[156]= $\{0.0000179695, 0.0000393953, 2.39109 \times 10^{-6}, 0.000143307\}$