Analysis of thiol reactivity data from *III3C* to determine open/close kinetics.

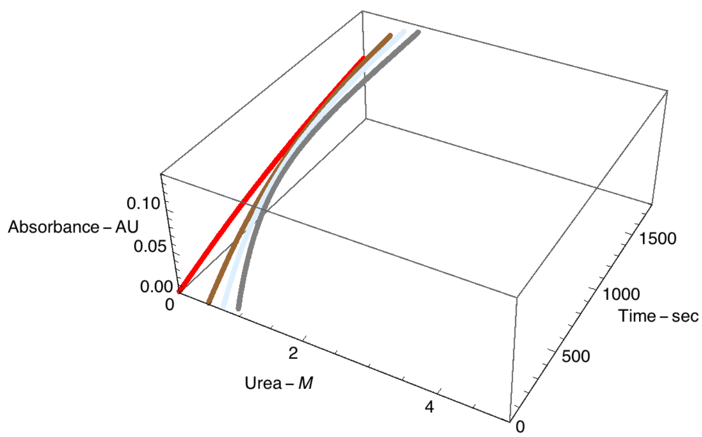
Spectrophotometer Data

Importing SpecData at various [Urea] in .txt format :

```
FilePath = "/Users/riddhishah/Google
    Drive/Riddhi's Stuff/Final Data Collection/Data/III3C_110513/";
ProteinConc = "10uM";
ProteinName = "III3C";
UreaConc = {0, 0.5, 0.75, 1};
DTNBconc = 100;
Nureas = Length[UreaConc];
Nreps = {1, 1, 1, 1};
Filenames = Table[FilePath <> ProteinConc <> ProteinName <>
    "_" <> ToString[DTNBconc] <> "uMDTNB_" <> ToString[UreaConc[[i]]] <>
    "M_R" <> ToString[j] <> ".txt", {i, Nureas}, {j, Nreps[[i]]}];
RawSpecData = Table[Drop[Import[Filenames[[i, j]], "Data"], 2],
   {i, Nureas}, {j, Nreps[[i]]}];
RawSpecDataF = Table[Cases[RawSpecData[[i, j]], {_, _?NumericQ}, ∞],
   {i, Nureas}, {j, Nreps[[i]]}];
SpecDataSize = Table[Length[RawSpecDataF[[i, 1]]], {i, Nureas}];
SpecData = Table[Flatten[
    {RawSpecDataF[[i, 1, k, 1]], Table[RawSpecDataF[[i, j, k, 2]], {j, Nreps[[i]]}]}],
   {i, Nureas}, {k, SpecDataSize[[i]]}];
NdatSpec = Table[Length[SpecData[[i]]], {i, Nureas}];
NrepsSpec =
  Table[Length[Select[SpecData[[i, NdatSpec[[i]]]], # ≠ 0 &]] - 1, {i, Nureas}];
SpecDataRep = Table[{SpecData[[i, j, 1]], SpecData[[i, j, k + 1]]},
   {i, Nureas}, {k, Nreps[[i]]}, {j, NdatSpec[[i]]}];
Specall1 = Flatten[Table[Flatten[{k, DTNBconc, UreaConc[[i]], SpecDataRep[[i, k, j]]}],
    {i, Nureas}, {k, Nreps[[i]]}, {j, NdatSpec[[i]]}], 1];
```

3D Plot of the imported Spec Data

```
ListPointPlot3D[Specall1[[All, All, 3 ;; 5]],
 PlotRange → {{0, 5}, {0, 1800}, {0, 0.14}},
 AxesLabel → {Urea - M , Time - sec , Absorbance - AU},
 PlotStyle -> {Red, Brown, LightBlue, Gray, Pink, Cyan, Magenta, Yellow,
   Green, Orange, Purple, Lighter[Purple, 0.5], Lighter[Pink, 0.5],
   Lighter[Blue, 0.5], Lighter[Black, 0.75], Lighter[Brown, 0.3]}]
```

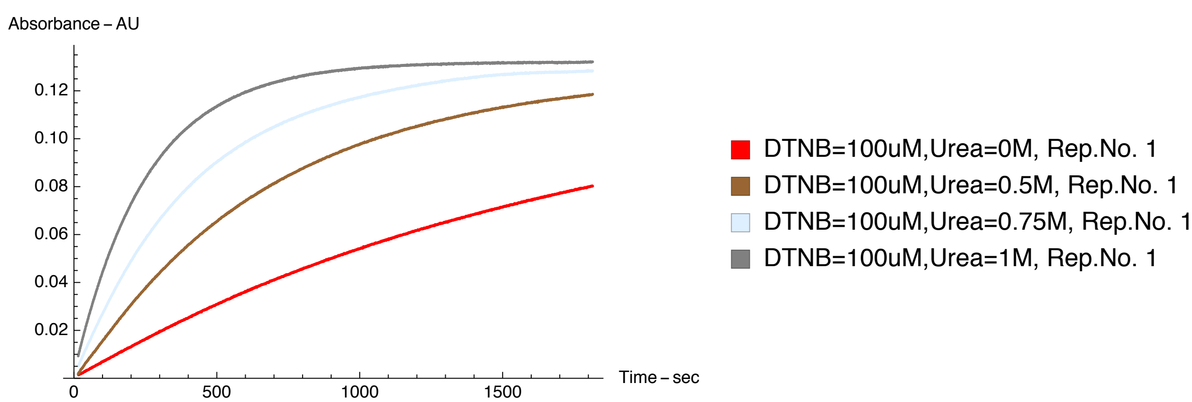


Addition of time lag into the imported SpecData

```
tLag = 15;
NtransientsSpec = Total[NrepsSpec];
NrowsSpec = Table[Length[Specall1[[i]]], {i, NtransientsSpec}];
SpecFinal = Table[
   Flatten[{Specall1[[i, j, 1 ;; 3]], Specall1[[i, j, 4]] + tLag, Specall1[[i, j, 5]]}],
   {i, NtransientsSpec}, {j, NrowsSpec[[i]]}];
SwatchLabelsSpec = Table["DTNB=" <> ToString[Specall1[[i, 1, 2]]] <>
     "uM,Urea=" <> ToString[Specall1[[i, 1, 3]]] <> "M, Rep.No. " <>
     ToString[Specall1[[i, 1, 1]]], {i, NtransientsSpec}];
```

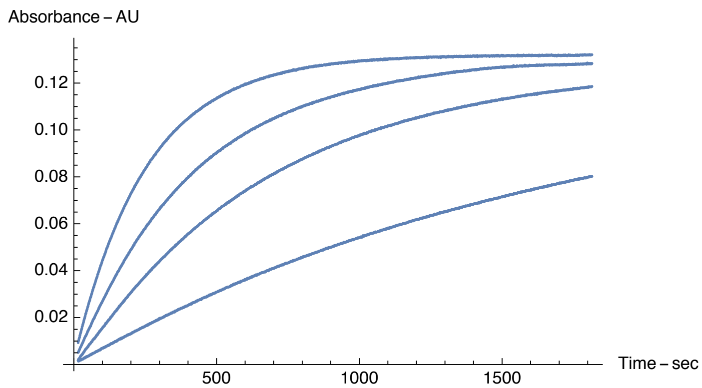
Plot of the edited SpecData

```
ListPlot[SpecFinal[[All, All, 4 ;; 5]], PlotRange → {0, Automatic},
 PlotLegends → SwatchLegend[SwatchLabelsSpec],
 AxesLabel → {Time - sec , Absorbance - AU},
 PlotStyle → {Red, Brown, LightBlue, Gray, Pink, Cyan, Magenta, Yellow,
   Green, Orange, Purple, Lighter[Purple, 0.5], Lighter[Pink, 0.5],
   Lighter[Blue, 0.5], Lighter[Black, 0.75], Lighter[Brown, 0.3]}]
```



Create a fittable dataset for the SpecData & Plot it -' FittableDataSpec'

```
NtransientsSpecall = Length[Specall1];
NdatSpecall = Table[Length[Specall1[[i]]], {i, NtransientsSpecall}];
FittableDataSpec = Flatten[Table[SpecFinal[[i, j, 2 ;; 5]],
     {i, 1, NtransientsSpecall}, {j, NdatSpecall[[i]]}], 1];
ListPlot[FittableDataSpec[[All, 3 ;; 4]], AxesLabel → {Time - sec , Absorbance - AU}]
```



Stop Flow Data

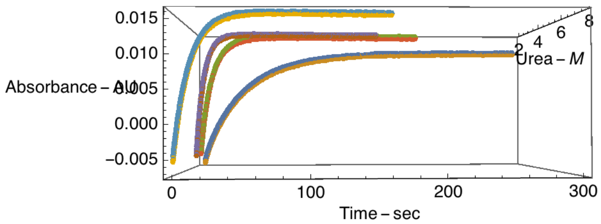
Importing SF Data at various [Urea] & [DTNB] i n .csv format

1. Importing SF Data from 100 uM DTNB :

```
FilePath = "/Users/riddhishah/Google
     Drive/Riddhi's Stuff/Final Data Collection/Data/III3C_110513/";
ProteinConc = "10uM";
ProteinName = "III3C";
DTNBconc1 = 100;
UreaConcSF1 = {2, 3, 4, 8};
NureasSF1 = Length[UreaConcSF1];
Filenames =
  Table[FilePath <> ProteinConc <> ProteinName <> "_" <> ToString[DTNBconc1] <>
     "uMDTNB_" <> ToString[UreaConcSF1[[i]]] <> "M.csv", {i, NureasSF1}];
SFData = Table[Drop[Drop[Import[Filenames[[i]], "Data"], 29], -4], {i, NureasSF1}];
NdatSF = Table[Length[SFData[[i]]], {i, NureasSF1}];
NrepsSF =
  Table[Length[Select[SFData[[i, NdatSF[[i]]]], # ≠ 0 &]] - 1, {i, NureasSF1}];
SFDataRep = Table[{SFData[[i, j, 1]], SFData[[i, j, k + 1]]},
    {i, NureasSF1}, {k, NrepsSF[[i]]}, {j, NdatSF[[i]]}];
SFall1 = Flatten[Table[Flatten[{k, DTNBconc1, UreaConcSF1[[i]], SFDataRep⟦i, k, j⟧}],
     {i, NureasSF1}, {k, NrepsSF⟦i⟧}, {j, NdatSF[[i]]}], 1];
NtransientsSF = Total[NrepsSF];
NrowsSF = Table[Length[SFall1[[i]]], {i, NtransientsSF}];
```

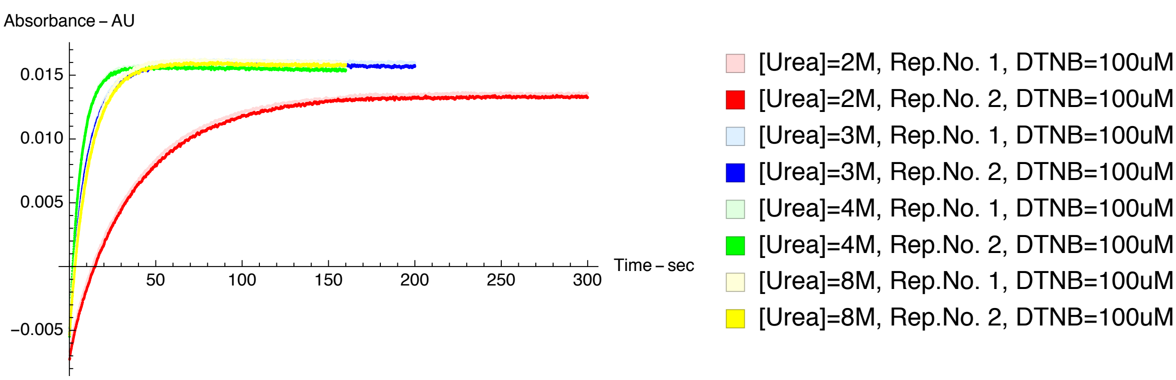
3D Plot of the imported 100uM DTNB SF Data

```
ListPointPlot3D[SFall1[[All, All, 3 ;; 5]],
 PlotRange → Full, AxesLabel → {Urea - M , Time - sec , Absorbance - AU}]
```



2D Plot of the imported 100uM DTNB SF Data

```
SwatchLabels = Table["[Urea]=" <> ToString[SFall1[[i, 1, 3]]] <>
    "M, Rep.No. " <> ToString[SFall1[[i, 1, 1]]] <> ", DTNB=" <>
    ToString[SFall1[[i, 1, 2]]] <> "uM", {i, NtransientsSF}];
ListPlot[SFall1[[All, All, 4 ;; 5]], PlotRange → Full,
 PlotLegends → SwatchLegend[SwatchLabels], AxesLabel → {Time - sec , Absorbance - AU},
 PlotStyle → {LightRed, Red, LightBlue, Blue, LightGreen, Green, LightYellow, Yellow}]
```

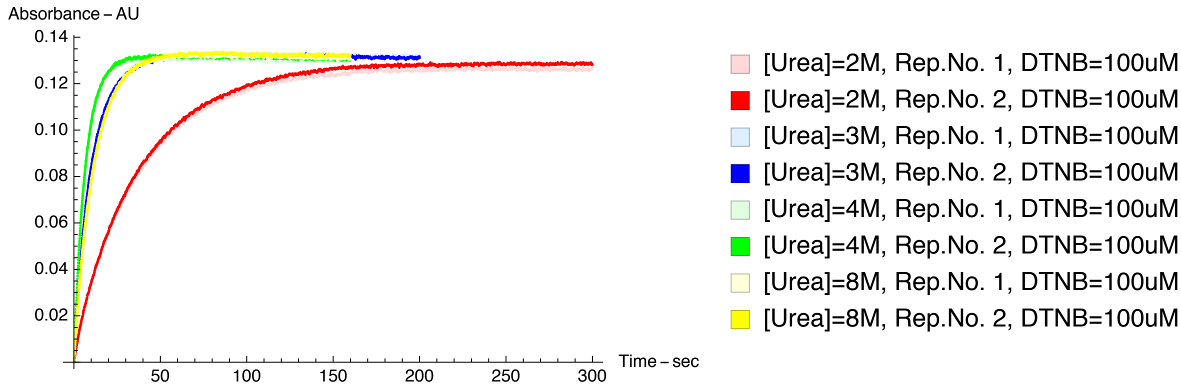


Ensuring that all SF Data for 100uM DTNB starts at time = 0

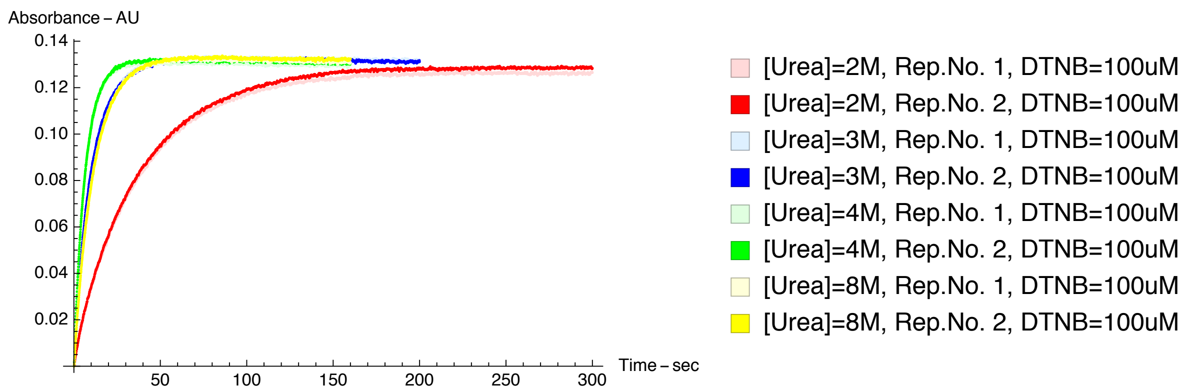
```
SFfinal1 =
  Table[Flatten[{SFall1[[i, j, 1 ;; 4]], 6.25 (SFall1[[i, j, 5]] - SFall1[[i, 1, 5]])}],
   {i, NtransientsSF}, {j, NrowsSF[[i]]}];
```

Plot of edited SF Data for 100uM DTNB

```
ListPlot[SFfinal1[[All, All, 4 ;; 5]], PlotRange → Automatic,
 PlotLegends → SwatchLegend[SwatchLabels], AxesLabel → {Time - sec , Absorbance - AU},
 PlotStyle → {LightRed, Red, LightBlue, Blue, LightGreen, Green, LightYellow, Yellow}]
```



ListPlot::prng : Value of option PlotRange -> {Automatic} is not All,
Full, Automatic, a positive machine number, or an appropriate list of range specifications. >>

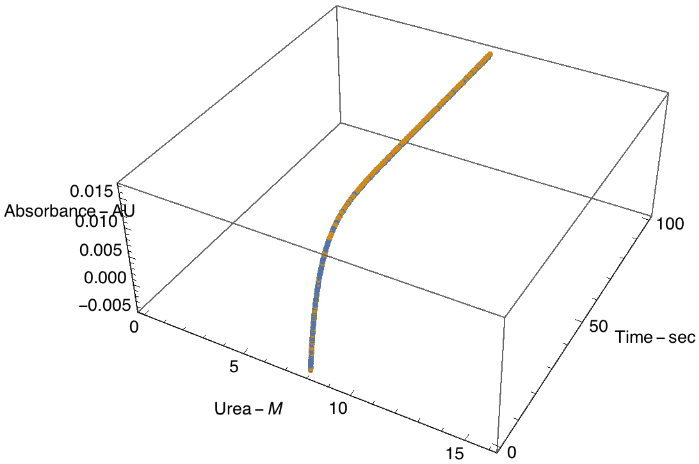


2. Importing SF Data from 195 uM DTNB :

```
(*Import data from stopped flow*)
FilePath = "/Users/riddhishah/Google
    Drive/Riddhi's Stuff/Final Data Collection/Data/III3C_110513/";
ProteinConc = "10uM";
ProteinName = "III3C";
DTNBconc2 = 195;
UreaConcSF2 = {8};
NureasSF2 = Length[UreaConcSF2];
Filenames =
  Table[FilePath <> ProteinConc <> ProteinName <> "_" <> ToString[DTNBconc2] <>
    "uMDTNB_" <> ToString[UreaConcSF2[[i]]] <> "M.csv", {i, NureasSF2}];
SFData2 = Table[Drop[Drop[Import[Filenames[[i]], "Data"], 29], -4], {i, NureasSF2}];
NdatSF2 = Table[Length[SFData2[[i]]], {i, NureasSF2}];
NrepsSF2 =
  Table[Length[Select[SFData2[[i, NdatSF2[[i]]]], # ≠ 0 &]] - 1, {i, NureasSF2}];
SFDataRep2 = Table[{SFData2[[i, j, 1]], SFData2[[i, j, k + 1]]},
   {i, NureasSF2}, {k, NrepsSF2[[i]]}, {j, NdatSF2[[i]]}];
SFall2 = Flatten[Table[Flatten[{k, DTNBconc2, UreaConcSF2[[i]], SFDataRep2[[
      i, k, j]]}], {i, NureasSF2}, {k, NrepsSF2[[i]]}, {j, NdatSF2[[i]]}], 1];
NtransientsSF2 = Total[NrepsSF2];
NrowsSF2 = Table[Length[SFall2[[i]]], {i, NtransientsSF2}];
```

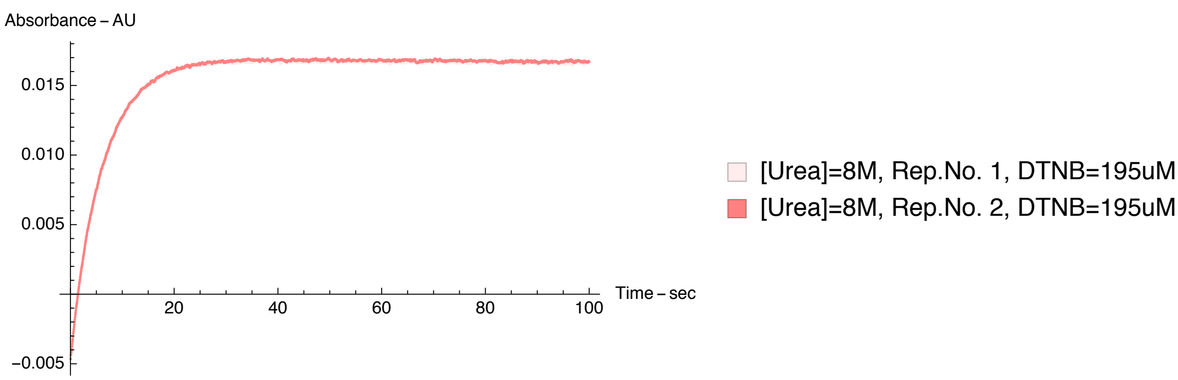
3D Plot of the imported 195uM DTNB SF Data

```
ListPointPlot3D[SFall2[[All, All, 3 ;; 5]], PlotRange → Full,
 AxesLabel → {Urea - M , Time - sec , Absorbance - AU}]
```



2D Plot of the imported 195uM DTNB SF Data

```
SwatchLabels2 = Table["[Urea]=" <> ToString[SFall2[[i, 1, 3]]] <>
    "M, Rep.No. " <> ToString[SFall2[[i, 1, 1]]] <> ", DTNB=" <>
    ToString[SFall2[[i, 1, 2]]] <> "uM", {i, NtransientsSF2}];
ListPlot[SFall2[[All, All, 4 ;; 5]], PlotRange → Full,
 PlotLegends → SwatchLegend[SwatchLabels2],
 AxesLabel -> {Time - sec , Absorbance - AU}, PlotStyle → {LightPink, Pink}]
```

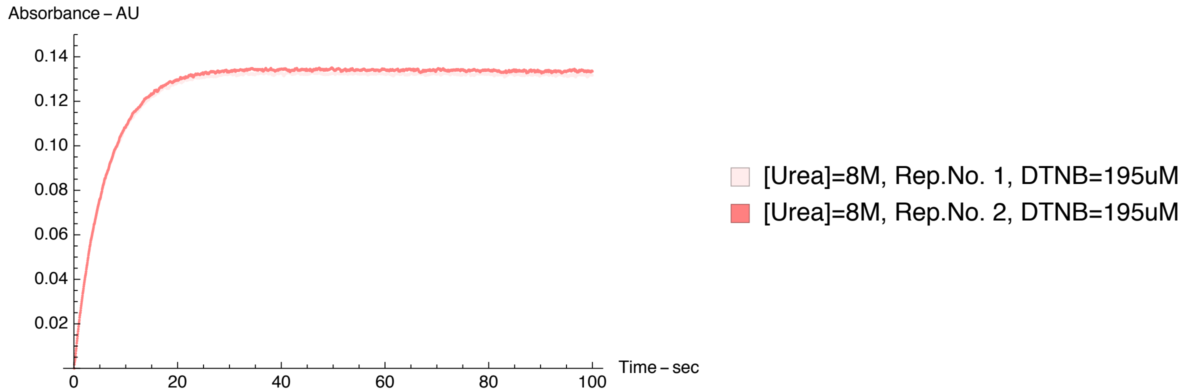


Ensuring that all SF Data for 195uM DTNB starts at time = 0

```
SFfinal2 =
  Table[Flatten[{SFall2[[i, j, 1 ;; 4]], 6.25 (SFall2[[i, j, 5]] - SFall2[[i, 1, 5]])}],
   {i, NtransientsSF2}, {j, NrowsSF2[[i]]}];
```

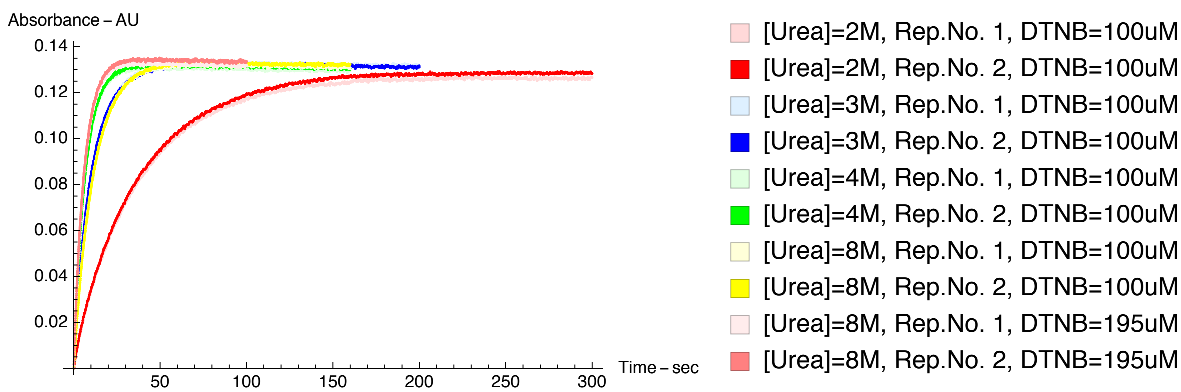
Plot of edited SF Data for 390uM DTNB

```
ListPlot[SFfinal2[[All, All, 4 ;; 5]],
 PlotRange → {0, 0.15}, PlotLegends → SwatchLegend[SwatchLabels2],
 PlotStyle → {LightPink, Pink}, AxesLabel → {Time - sec , Absorbance - AU}]
```



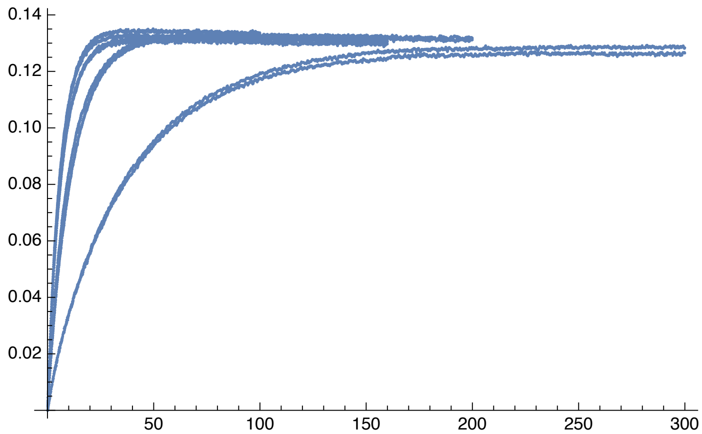
Combine and Plot all SFData - ' SFAllData'

```
SFAllData = Join[SFfinal1, SFfinal2];
NdatSFAllData = Length[SFAllData];
SwatchLabelsall = Table["[Urea]=" <> ToString[SFAllData[[i, 1, 3]]] <>
    "M, Rep.No. " <> ToString[SFAllData[[i, 1, 1]]] <> ", DTNB=" <>
    ToString[SFAllData[[i, 1, 2]]] <> "uM", {i, NdatSFAllData}];
ListPlot[SFAllData[[All, All, 4 ;; 5]], PlotRange → Full,
 PlotLegends → SwatchLegend[SwatchLabelsall],
 PlotStyle → {LightRed, Red, LightBlue, Blue, LightGreen, Green, LightYellow,
   Yellow, LightPink, Pink}, AxesLabel → {Time - sec , Absorbance - AU}]
```



Create a fittable dataset for SFData & Plot it - ' FittableDataSF'

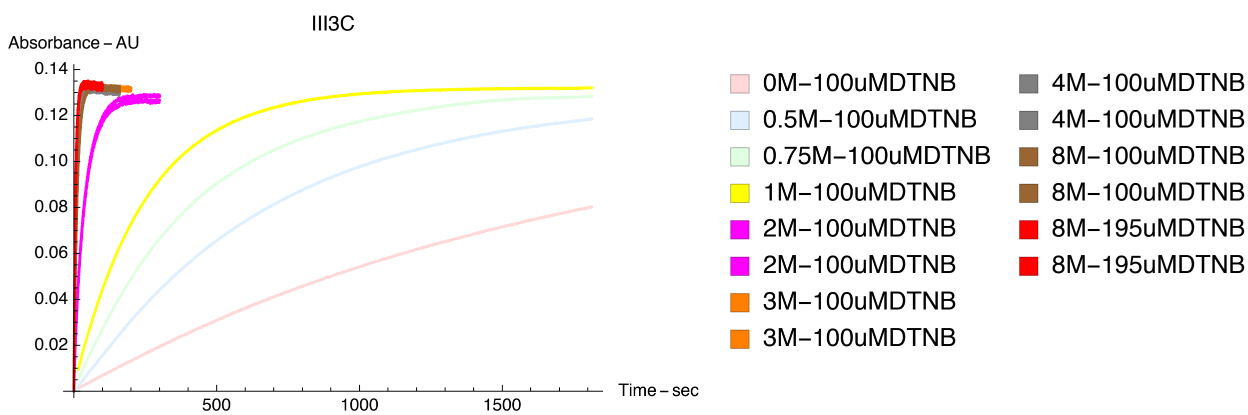
```
Ntransients = Length[SFAllData];
NdatAll = Table[Length[SFAllData[[i]]], {i, Ntransients}];
FittableDataSF =
  Flatten[Table[SFAllData[[i, j, 2 ;; 5]], {i, Ntransients}, {j, NdatAll[[i]]}], 1];
ListPlot[FittableDataSF[[All, 3 ;; 4]]]
```



Data fitting

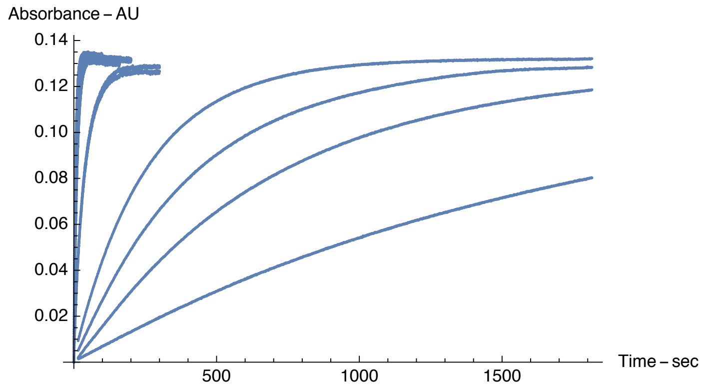
Combine Spec & SF Data :

```
AllData = Join[SpecFinal, SFAllData];
NtransientsAll = Length[AllData];
NdatAllData = Length[AllData];
LabelsAllData = Table[ToString[AllData[[i, 1, 3]]] <> "M-" <>
     ToString[AllData[[i, 1, 2]]] <> "uM" <> "DTNB", {i, NdatAllData}];
III3Cplot = ListPlot[AllData[[All, All, 4 ;; 5]],
   AxesLabel → {Time - sec , Absorbance - AU}, PlotLabel → "III3C",
   PlotStyle → {LightRed, LightBlue, LightGreen, Yellow, Magenta,
     Magenta, Orange, Orange, Gray, Gray, Brown, Brown, Red, Red},
   PlotLegends → SwatchLegend[LabelsAllData]]
```



Create a fittable dataset for Spec & SF Data - ' AllFittableData' :

```
AllFittableData = Join[FittableDataSF, FittableDataSpec];
ListPlot[AllFittableData[[All, 3 ;; 4]], AxesLabel → {Time - sec , Absorbance - AU}]
```



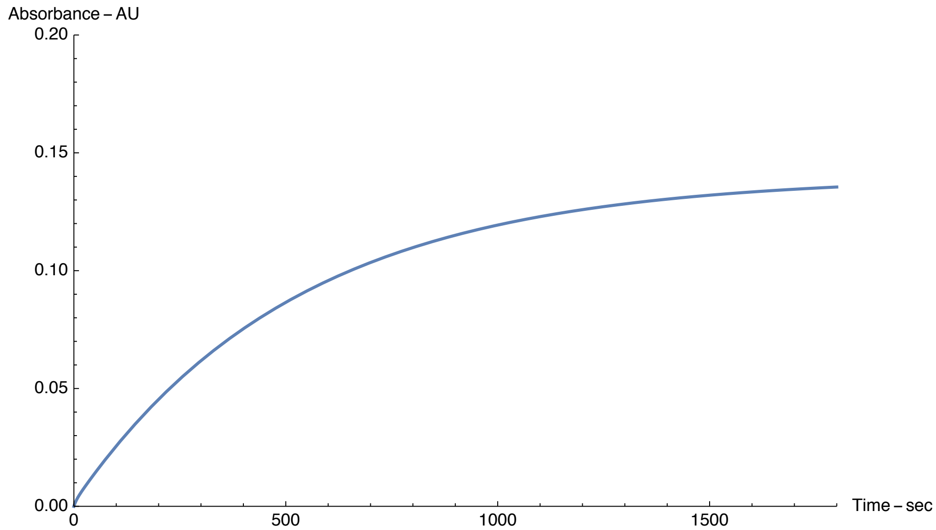

```
Export["AllfittableDAta3C.csv", AllFittableData, "Data"];
```

Model for fitting the data :

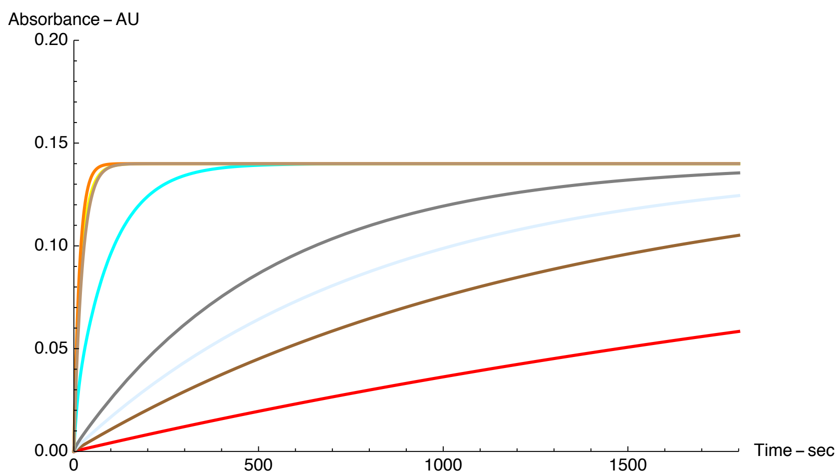
```
Signal[Rconc_, Uconc_, t_] := Module[{kop, kcl, kchem},

  kop = Exp[Log[kop0] + mop * Uconc];
  kcl = Exp[Log[kcl0] + mcl * Uconc];
  kchem = Exp[Log[kchem0] - 0.1 * Uconc];
  RateMatrix = ( -kop   kcl                          0
                 kop    -kcl - kchem * Rconc * 10^-6  0
                 0      kchem * Rconc * 10^-6        0 );
  λvector = Eigenvalues[RateMatrix];
  Bmatrix = Transpose[Eigenvectors[RateMatrix]];
  InvBmatrix = Inverse[Bmatrix];
  ExpΛmatrix = DiagonalMatrix[Exp[λvector t]];
  Keq = kop / kcl;
  P0 = {1 / (1 + Keq), Keq / (1 + Keq), 0};
  P = Bmatrix .ExpΛmatrix .InvBmatrix.P0;
  P[[3]] AbsMax]
```

Simulate fit of ' Combined SF & Spec Data to obtain the initial kop, kcl, mop, mcl, kchem & AbsMax values.

```
kop0 = 0.0009;
kcl0 = 0.2;
mcl = -0.9;
mop = 1.4;
kchem0 = 1000;
AbsMax = 14 000 × 10^-5;

Plot[Signal[100, 1, t], {t, 0, 1800},
 PlotRange → {{0, 1800}, {0, 0.2}}, AxesLabel → {Time - sec , Absorbance - AU}]
```




```
Plot[{Signal[100, 0, t], Signal[100, 0.5, t], Signal[100, 0.75, t], Signal[100, 1, t],
  Signal[100, 2, t], Signal[100, 3, t], Signal[100, 4, t], Signal[100, 8, t]},
 {t, 0, 1800}, PlotRange → {{0, 1800}, {0, 0.2}},
 AxesLabel → {Time - sec , Absorbance - AU},
 PlotStyle → {Red, Brown, LightBlue, Gray, Cyan, Yellow, Orange, Lighter[Brown, 0.3]}]
```



Global fitting o f the data to obtain the initial kop, kcl, mop, mcl, kchem & AbsMax values.

```
Unset[{kop0, kcl0, mcl, mop, kchem0, AbsMax, Rconc, Uconc, t}];
kop0init = 0.001;
kcl0init = 0.3;
mclinit = -1.00;
mopinit = 1.00;
kchem0init = 1000;
AbsMaxinit = 13 000 × 10^-5;
Dynamic[{ev, st, kop0, kcl0, mcl, mop, kchem0, AbsMax}]
ev = 0; st = 0;
GlobalFit = NonlinearModelFit[AllFittableData, {Signal[Rconc, Uconc, t]
     (*,mcl<0&&mop>0&&kop0>0&&kcl0>0&&kchem>0)*)},
    {{kop0, kop0init}, {kcl0, kcl0init}, {mcl, mclinit}, {mop, mopinit},
     {kchem0, kchem0init}, {AbsMax, AbsMaxinit}}, {Rconc, Uconc, t},
    MaxIterations → 1000(*{Method→"NMinimize",Method->"SimulatedAnnealing"}*)];
```

Parameter listing & analysis

Peek at best fit parameter values without doing error analysis

```
GlobalFit["BestFitParameters"]
```

```
{kop0 → 0.00189692, kcl0 → 0.549814, mcl → -0.779821,
 mop → 1.53215, kchem0 → 2074.45, AbsMax → 0.131449}
```

```
GlobalFit["ParameterConfidenceIntervalTable"]
```

Estimate standard error, t-statistic & P-values for each parameter (slow!)

```
Unset[{kop0, kcl0, mcl, mop, kchem0, AbsMax}];
GlobalFit["ParameterTable"]
```

Compute parameter correlation coefficients (slow!) Magnitudes > ± 0.9 indicate pairs parameters for whom changing one parameter value can be compensated for by a change in the other, with no reduction in goodness of fit. In other words, the values of both parameters are poorly determined, regardless of their estimated standard errors.

```
Unset[{kop0, kcl0, mcl, mop, kchem0, AbsMax}];
TableForm[Round[GlobalFit["CorrelationMatrix"], 0.01],
 TableHeadings → {{"kop0", "kcl0", "mcl", "mop", "kchem0", "AbsMax"},
   {"kop0", "kcl0", "mcl", "mop", "kchem0", "AbsMax"}}]
```

	kop0	kcl0	mcl	mop	kchem0	AbsMax
kop0	1.	1.	-0.79	-0.86	0.07	-0.08
kcl0	1.	1.	-0.79	-0.85	0.12	-0.09
mcl	-0.79	-0.79	1.	0.99	0.1	-0.02
mop	-0.86	-0.85	0.99	1.	0.06	-0.02
kchem0	0.07	0.12	0.1	0.06	1.	-0.32
AbsMax	-0.08	-0.09	-0.02	-0.02	-0.32	1.

Determine the ‘RSquared’ value for the globalfit. The closer the result is to one the better the fit.

```
Unset[{kop0, kcl0, mcl, mop, kchem0, AbsMax}];
GlobalFit["RSquared"]
```

```
0.999776
```

Determine the ‘BIC’ value for the fits. This parameter checks the model. It punishes you for added parameters in the model.

```
Unset[{kop0, kcl0, mcl, mop, kchem0, AbsMax}];
GlobalFit["BIC"]
```

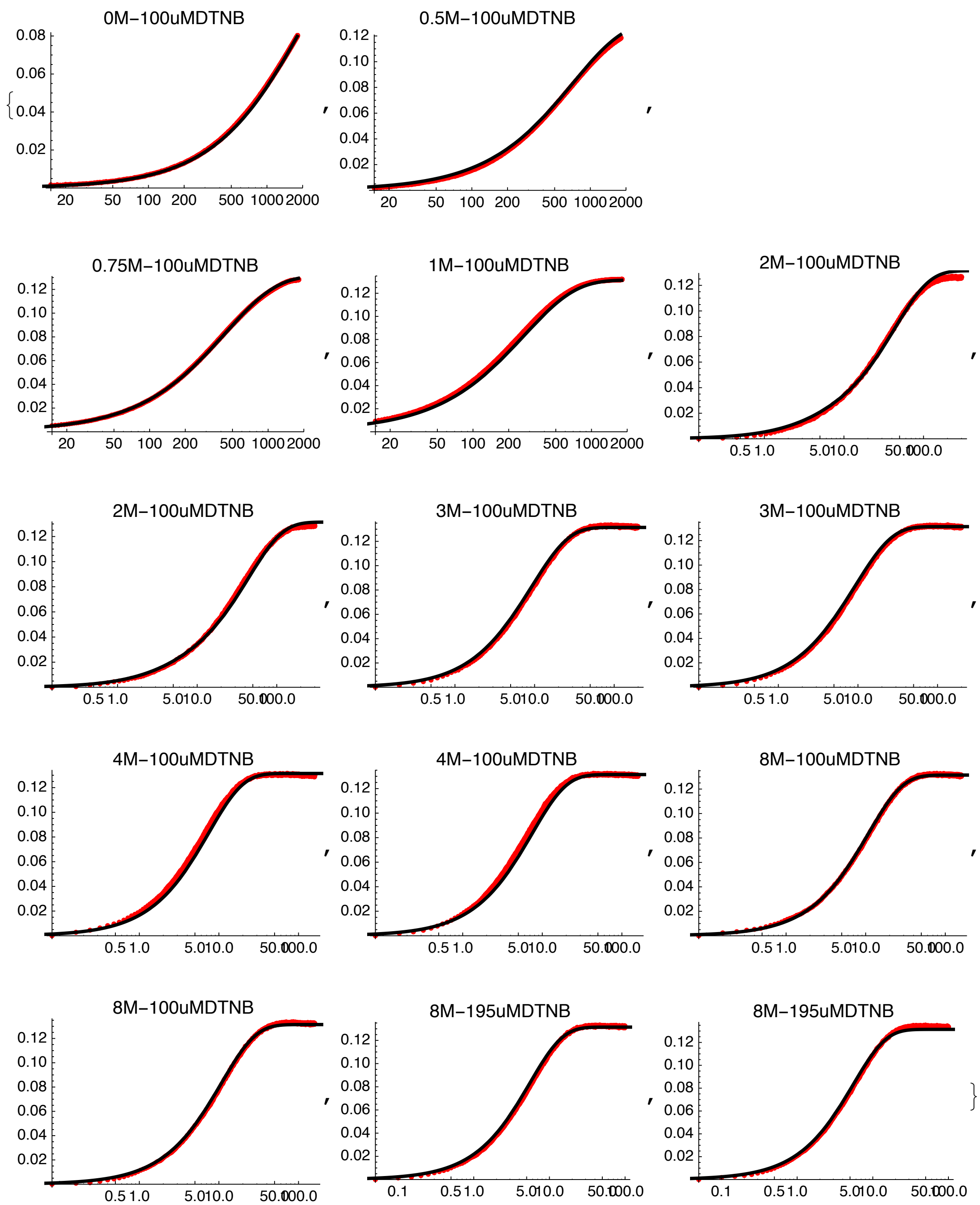
```
-268 017.
```

Apply best fit parameter values to each parameter variable so that they can be used for further analysis

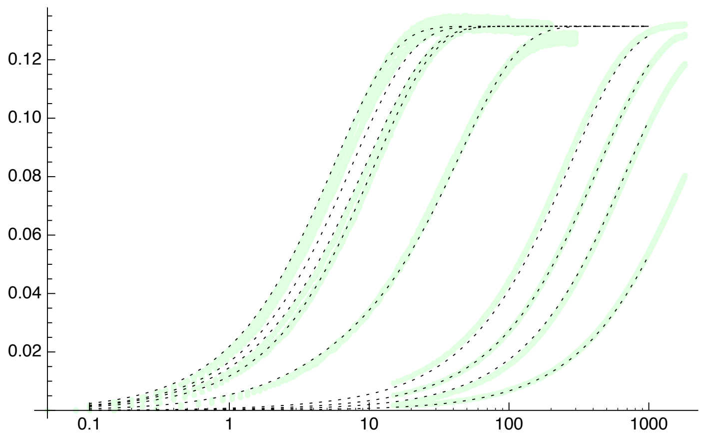
```
kop0 = GlobalFit["BestFitParameters"][[1, 2]];
kcl0 = GlobalFit["BestFitParameters"][[2, 2]];
mcl = GlobalFit["BestFitParameters"][[3, 2]];
mop = GlobalFit["BestFitParameters"][[4, 2]];
kchem0 = GlobalFit["BestFitParameters"][[5, 2]];
AbsMax = GlobalFit["BestFitParameters"][[6, 2]];
```

Plot data and fitted Signal(t) transients to visualize goodness of fit

```
FitPlot = Table[
   LogLinearPlot[Signal[AllData[[i, 1, 2]], AllData[[i, 1, 3]], t], {t, 0.001, 1800},
    PlotStyle → {Black, Thick}, PlotRange → Full], {i, NtransientsAll}];
Table[Show[ListLogLinearPlot[AllData[[i, All, 4 ;; 5]],
   PlotLabel → LabelsAllData[[i]], PlotRange → Full, PlotStyle → {Red, Thick}],
  FitPlot[[i]]], {i, NtransientsAll}]
```



```
FitPlot =
  LogLinearPlot[{Table[Signal[100, Uconc, t], {Uconc, {0, 0.5, 0.75, 1, 2, 3, 4, 8}}],
    Signal[195, 8, t]}, {t, .1, 1000}, PlotStyle → {{Dotted, Black}}];
DataPlot = ListLogLinearPlot[AllFittableData[[All, 3 ;; 4]],
   PlotRange → All, PlotStyle → {{LightGreen}}];
Show[DataPlot, FitPlot]
```



```
FitPlots = Table[Plot[Signal[AllData[[i, 1, 2]], AllData[[i, 1, 3]], t], {t, 0, 1800},
    PlotStyle → {Black, Thick}, PlotRange → Full], {i, NtransientsAll}];
Table[Show[ListPlot[AllData[[i, All, 4 ;; 5]], PlotRange → {Automatic},
   PlotLabel → LabelsAllData[[i]]], FitPlots[[i]]], {i, NtransientsAll}]
```

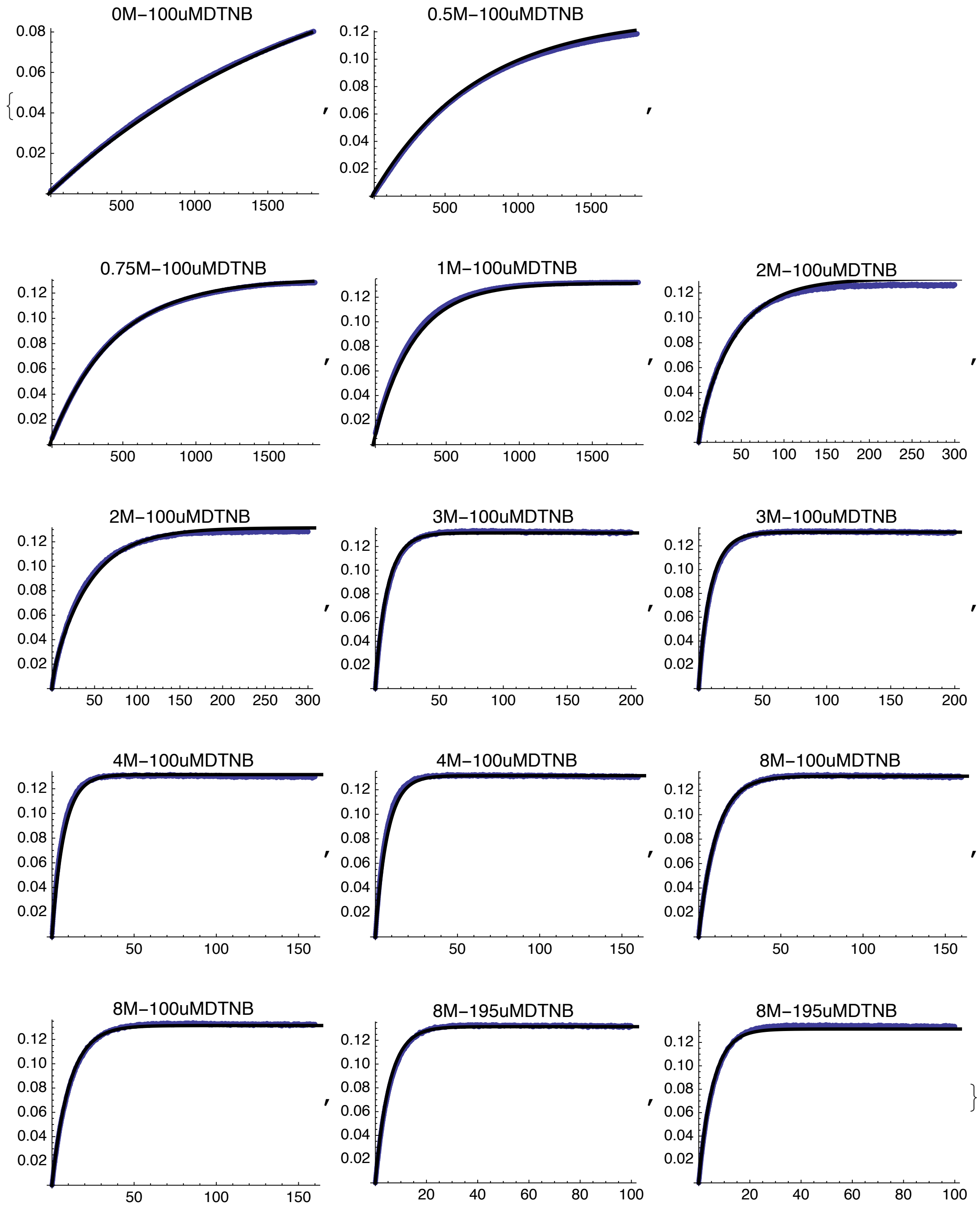
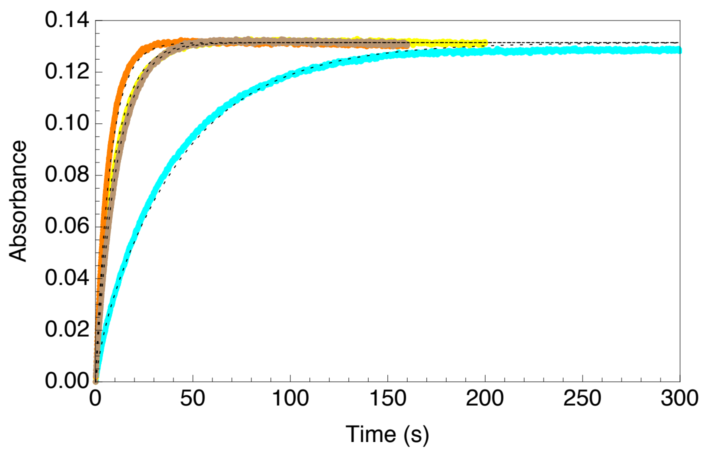
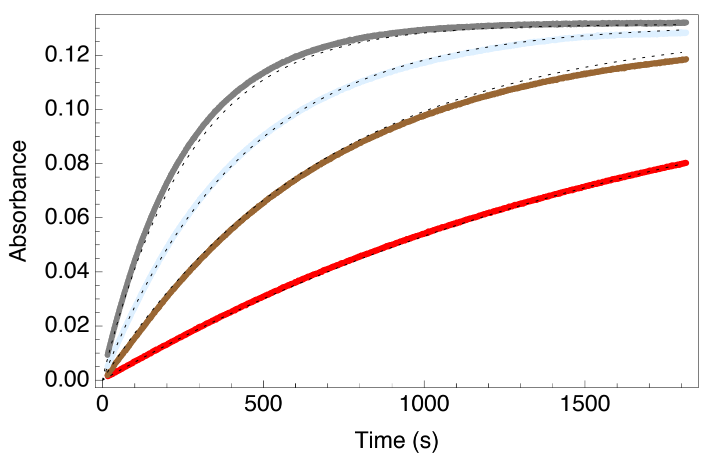


Figure for research paper

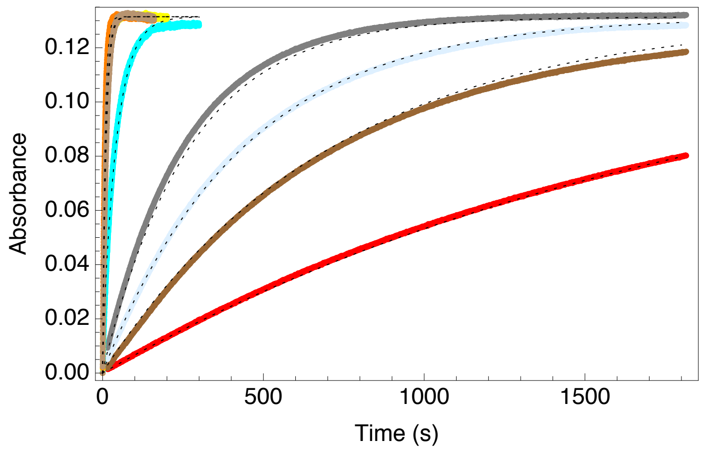
```
Forpaper3CFig8SF1 = Drop[SFAllData, {1, 6, 2}];
Forpaper3CFig8SF = Drop[Forpaper3CFig8SF1, -3];
Forpaper3CSF = Forpaper3CFig8SF;
NdatForpaper3CSF = Length[Forpaper3CSF];
SwatchLabelsForpaper3CSF =
  Table[ToString[Forpaper3CSF[[i, 1, 3]]] <> "M ", {i, 4}];
Forpaper3CSFfig = ListPlot[Forpaper3CSF[[1 ;; 4, All, 4 ;; 5]],
   PlotRange → {{0, 300}, {0, 0.14}}, (*PlotLegends→Placed[SwatchLegend[
      SwatchLabelsForpaper3CSF,LegendLayout→"Row"],{{0.4,-0.2},{0.4,-0.2}}],*)
   PlotStyle → {Cyan, Yellow, Orange, Lighter[Brown, 0.3], Lighter[Purple, 0.5],
     Lighter[Blue, 0.5], Lighter[Brown, 0.3], Red, Blue, Yellow},
   AxesLabel → {Time - sec , Absorbance - AU}, Frame → True,
   FrameLabel → {"Time (s)", "Absorbance"(*,"FNIII 3C Stopflow data "*)},
   LabelStyle → {(FontFamily → "Arial"), 12},
   FrameTicks → {Automatic, Automatic, None, None}];
FitPlotsForpaper3CSF = Plot[{Signal[Forpaper3CSF[[1, 1, 2]], Forpaper3CSF[[1, 1, 3]],
     t], Signal[Forpaper3CSF[[2, 1, 2]], Forpaper3CSF[[2, 1, 3]], t],
    Signal[Forpaper3CSF[[3, 1, 2]], Forpaper3CSF[[3, 1, 3]], t],
    Signal[Forpaper3CSF[[4, 1, 2]], Forpaper3CSF[[4, 1, 3]], t]}, {t, 0, 300},
   PlotStyle → {{Dotted, Black}}(*,{Dashing[Large],Black},{Dashed,Black},
    {Dashing[Tiny],Black},{DotDashed,Black},{Dashing[{Large}],Black},
    {Dashing[{Small,Large}],Black}}*), PlotRange → Full(*,PlotLegends→
    Placed[LineLegend[{"Fit"},LegendLayout→"Row"],{{0.88,0.01},{0.4,0.01}}]*)];

FigIII3CSF = Show[Forpaper3CSFfig, FitPlotsForpaper3CSF]
```



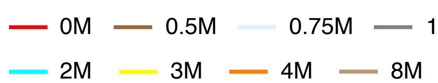
```
Forpaper3CSpec = SpecFinal;
NdatForpaper3CSpec = Length[Forpaper3CSpec];
SwatchLabelsForpaper3CSpec =
  Table[ToString[Forpaper3CSpec[[i, 1, 3]]] <> "M", {i, 4}];
Forpaper3CSpecfig = ListPlot[Forpaper3CSpec[[1 ;; 4, All, 4 ;; 5]], PlotRange → Full,
   (*PlotLegends→Placed[SwatchLegend[SwatchLabelsForpaper2CSpec,
      LegendLayout→"Row"],{{0.8,-0.15},{0.8,-0.15}}],*)
   PlotStyle → {Red, Brown, LightBlue, Gray, Pink, Cyan, Magenta, Yellow,
     Green, Orange, Purple, Lighter[Purple, 0.5], Lighter[Pink, 0.5],
     Lighter[Blue, 0.5], Lighter[Black, 0.75], Lighter[Brown, 0.3]},
   AxesLabel → {Time - sec , Absorbance - AU}, Frame → True,
   FrameLabel → {"Time (s)", "Absorbance"(*,"FNIII 3C Spectrophotometer data"*)},
   LabelStyle → {(FontFamily → "Arial"), 12},
   FrameTicks → {Automatic, Automatic, None, None}];
FitPlotsForpaper3CSpec = Plot[
   {Signal[Forpaper3CSpec[[1, 1, 2]], Forpaper3CSpec[[1, 1, 3]], t],
    Signal[Forpaper3CSpec[[2, 1, 2]], Forpaper3CSpec[[2, 1, 3]], t],
    Signal[Forpaper3CSpec[[3, 1, 2]], Forpaper3CSpec[[3, 1, 3]], t],
    Signal[Forpaper3CSpec[[4, 1, 2]], Forpaper3CSpec[[4, 1, 3]], t]}, {t, 0, 1800},
   PlotStyle → {{Dotted, Black}}(*,{Dashing[Large],Black},{Dashed,Black},
    {Dashing[Tiny],Black},{DotDashed,Black},{Dashing[{Large}],Black},
    {Dashing[{Small,Large}],Black}}*), PlotRange → Full(*,PlotLegends→
    Placed[LineLegend[{"Fit"},LegendLayout→"Row"],{{0.3,-0.25},{1,-0.25}}]*)];

FigIII3CSpec = Show[Forpaper3CSpecfig, FitPlotsForpaper3CSpec]
```



```
FigforpaperIII3CAll = Show[Forpaper3CSpecfig,
  Forpaper3CSFfig, FitPlotsForpaper3CSpec, FitPlotsForpaper3CSF]
```

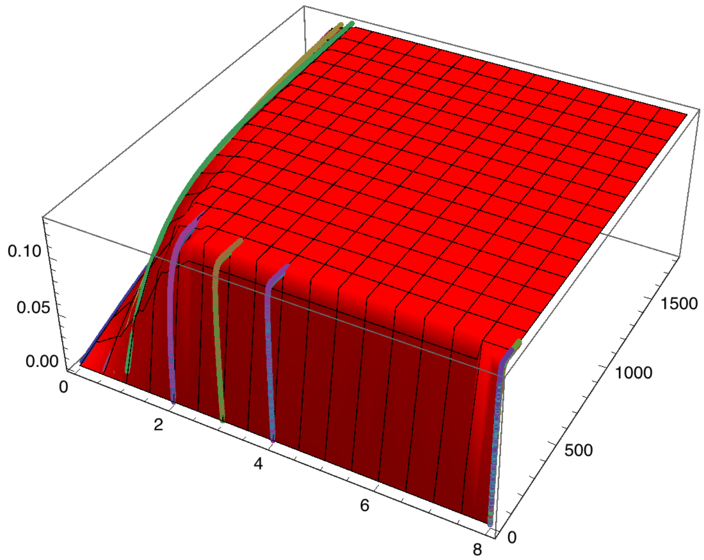



```
FigIII3Clegends = LineLegend[{Directive[Red, Thick],
   Directive[Brown, Thick], Directive[LightBlue, Thick], Directive[Gray, Thick],
   Directive[Cyan, Thick], Directive[Yellow, Thick], Directive[Orange, Thick],
   Directive[Lighter[Brown, 0.3], Thick], Directive[Orange, Thick],
   Directive[Lighter[Purple, 0.5], Thick], Directive[Lighter[Blue, 0.5], Thick],
   Directive[Lighter[Brown, 0.3], Thick], {Dotted, Black}},
  {SwatchLabelsForpaper3CSpec[[1]], SwatchLabelsForpaper3CSpec[[2]],
   SwatchLabelsForpaper3CSpec[[3]], SwatchLabelsForpaper3CSpec[[4]],
   SwatchLabelsForpaper3CSF[[1]], SwatchLabelsForpaper3CSF[[2]],
   SwatchLabelsForpaper3CSF[[3]], SwatchLabelsForpaper3CSF[[4]]},
  LegendLayout → "Row", LabelStyle → {(FontFamily → "Arial"), 12}]
```



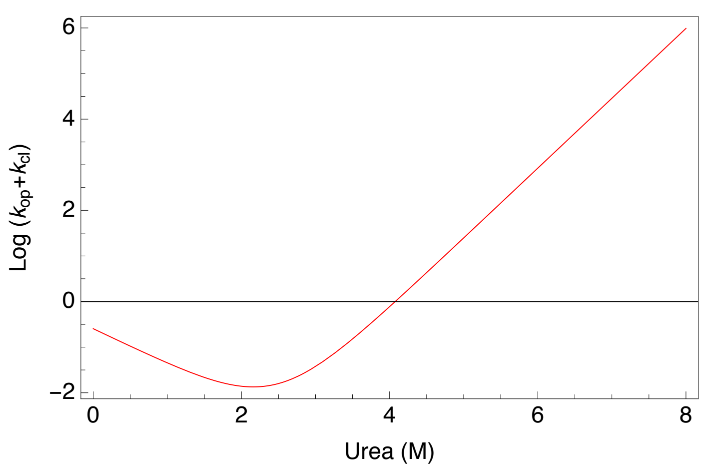
Plot data & fits in three dimensions (Signal vs. [urea] & t)

```
tMax = 1800;
FitPlots = Plot3D[Signal[100, i, t],
   {i, 0, 8}, {t, 0, tMax}, PlotStyle → Red, PlotRange → Full];
DataPlots = ListPointPlot3D[AllData[[All, All, 3 ;; 5]]];
Show[FitPlots, DataPlots]
```



Plot simulated chevron plot (ln(kop + kcl) vs. [urea])

```
kop[Uconc_] := Exp[Log[kop0] + mop * Uconc];
kcl[Uconc_] := Exp[Log[kcl0] + mcl * Uconc];
kchem[Uconc_] := Exp[Log[kchem0] - 0.1 * Uconc]
III3CChevronPlot = Plot[Log[kop[Uconc] + kcl[Uconc]], {Uconc, 0, 8},
  Frame → True, FrameLabel → {"Urea (M)", "Log (kop+kcl)", None},
  LabelStyle → {(FontFamily → "Arial"), 12},
  FrameTicks → {Automatic, Automatic, None, None}, PlotStyle → Red, PlotRange → All]
```



Compute β_T

```
βT = mcl / (mcl - mop)
```

```
0.337298
```

Compute m_{eq}

```
meq = .6 (mcl - mop)
```

```
-1.38718
```

Compute $K_{eq(cl)}$at [urea] = 0

```
Keq = kcl0 / kop0
289.845
```

Compute $K_{eq(op)}$at [urea] = 0

```
Keqop = kop0 / kcl0
0.00345012
```

Compute $C^{1/2}$

```
Log[Keqop] / (mcl - mop)
2.45218
```

Compute ΔG_{op} at [urea] = 0

```
DeltaGop = .6 Log[kcl0 / kop0]
3.40161
```

Compute number of residues exposed in TS for opening

```
meq = 374 + .11 ASA;
ASA = -907 + 93 Nres;
ResidueNo = Solve[meq == -1.38, Nres]
{{Nres → -26.9413}}
```

Compute folded/closed state lifetime (in seconds)

```
Closedlifetime = 1 / kop0
527.17
```

Compute folded/closed state halflifetime (in seconds)

```
Closedhalflifetime = 0.693 / kop0
365.329
```

Compute open state lifetime (in seconds)

```
Openlifetime = 1 / kcl0
1.8188
```

Compute open state halflifetime (in seconds)

```
Openhalflifetime = 0.693 / kcl0
```

```
1.26043
```

Error propogation in computed parameters

	Estimate	Standard Error	t-Statistic	P-Value
kop0	0.00189692	0.000024229	78.2914	$5.2202203229\times10^{-1203}$
kcl0	0.549814	0.0101957	53.9261	$1.7930696462\times10^{-602}$
mcl	−0.779821	0.00957587	−81.436	$2.2847892514\times10^{-1291}$
mop	1.53215	0.00660216	232.067	$4.6959320602\times10^{-6452}$
kchem0	2074.45	2.24013	926.041	$1.3967447643\times10^{-20568}$
AbsMax	0.131449	0.0000140488	9356.57	$4.8787523498\times10^{-47706}$

```
R1 = 1 / x;
R2 = 1 / y;
R3 = y / x;
R4 = .6 Log[y / x];
R5 = .6 (y - x);
R6 = 0.693 / x;
R7 = 0.693 / y;
R8 = x / y;
R9 = y / (y - x);
deltax = 0.000024;
deltay = 0.010195696724145418`;
deltau = 0.006602159099735208`;
deltav = 0.009575867475265872`;
```

```
ErrorClosed = Sqrt[Power[D[R1, x] * deltax, 2]]
```

$$0.000024 \sqrt{\frac{1}{x^4}}$$

```
ErrorHalfClosed = Sqrt[Power[D[R6, x] * deltax, 2]]
```

$$0.000016632 \sqrt{\frac{1}{x^4}}$$

```
ErrorOpen = Sqrt[Power[D[R2, y] * deltay, 2]]
```

$$0.0101957 \sqrt{\frac{1}{y^4}}$$

```
ErrorHalfOpen = Sqrt[Power[D[R7, y] * deltay, 2]]
```

$$0.00706562 \sqrt{\frac{1}{y^4}}$$

```
ErrorKeq = Sqrt[Power[D[R3, x] * deltax, 2] + Power[D[R3, y] * deltay, 2]]
```

$$\sqrt{\frac{0.000103952}{x^2}+\frac{5.76\times 10^{-10}\,y^2}{x^4}}$$

```
ErrorKeqop = Sqrt[Power[D[R8, x] * deltax, 2] + Power[D[R8, y] * deltay, 2]]
```

$$\sqrt{\frac{0.000103952\,x^2}{y^4}+\frac{5.76\times 10^{-10}}{y^2}}$$

```
ErrordeltaG = Sqrt[Power[D[R4, x] * deltax, 2] + Power[D[R4, y] * deltay, 2]]
```

$$\sqrt{\frac{2.0736\times 10^{-10}}{x^2}+\frac{0.0000374228}{y^2}}$$

```
Errormeq = Sqrt[Power[D[R5, x] * deltax, 2] + Power[D[R5, y] * deltay, 2]]
```

0.00611743

```
ErrorBT = Sqrt[Power[D[R9, x] * deltax, 2] + Power[D[R9, y] * deltay, 2]]
```

$$\sqrt{\left(\frac{5.76\times 10^{-10}\,y^2}{(-x+y)^4}+0.000103952\left(-\frac{y}{(-x+y)^2}+\frac{1}{-x+y}\right)^2\right)}$$

1. Closed state lifetime (in seconds)

```
x = kop0;
y = kcl0;
```

$$\text{ErrorClosed} = 0.000024\grave{}\ \sqrt{\frac{1}{x^4}}$$

6.66979

2. Open state lifetime (in seconds)

$$\text{ErrorOpen} = 0.010195696724145418\grave{}\ \sqrt{\frac{1}{y^4}}$$

0.0337276

3. $K_{eq(cl)}$ at[urea] = 0

$$\sqrt{\left(\frac{0.00010395223169074 96\grave{}}{x^2}+\frac{5.760000000000001\grave{}*\hat{}-10\ y^2}{x^4}\right)}$$

6.5067

4. ΔG_{op} at[urea] = 0

$$\sqrt{\left(\frac{2.0735999999999997\grave{}*\hat{}-10}{x^2} + \frac{0.00003742280340866985\grave{}}{y^2}\right)}$$

0.0134693

5. meq

Errormeq

0.00611743

6. Closed state halflife (in seconds)

$$\text{ErrorHalfClosed} = 0.000016632\grave{}\sqrt{\frac{1}{x^4}}$$

4.62217

7. Open state halflife (in seconds)

$$\text{ErrorHalfOpen} = 0.007065617829832774\grave{}\sqrt{\frac{1}{y^4}}$$

0.0233732

8. $K_{eq(op)}$ at[urea] = 0

$$\text{ErrorKeqop} = \sqrt{\left(\frac{0.0001039522316907496\grave{}\ x^2}{y^4} + \frac{5.760000000000001\grave{}*\hat{}-10}{y^2}\right)}$$

0.0000774512

9. β_T

$\text{Error}\beta_T$ =

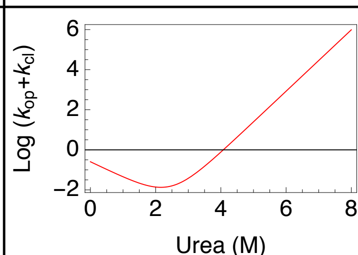
$$\sqrt{\left(\frac{5.760000000000001\grave{}*\hat{}-10\ y^2}{(-x+y)^4} + 0.0001039522316907496\grave{}\left(-\frac{y}{(-x+y)^2} + \frac{1}{-x+y}\right)^2\right)}$$

0.0000779885

Convert to output form & export as .pdf

1. Output form for all the parameters & associated statistics

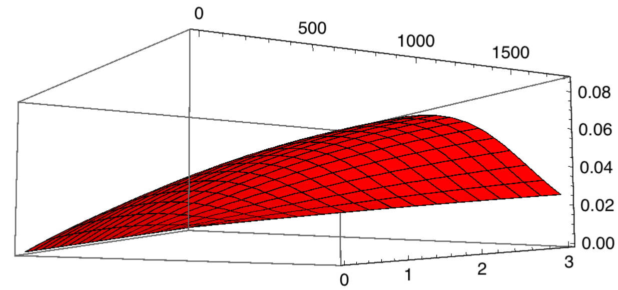
```
Output3C = Grid[{{"Parameters at 0M Urea", "Value", "Error"},
   {"kop(s^-1)", kop0, deltax}, {"kcl (s^-1)", kcl0, deltay},
   {"Closed state half-life(s)", Closedhalflifetime, ErrorHalfClosed},
   {"Open state half-life(s)", Openhalflifetime, ErrorHalfOpen},
   {"mcl", mcl, deltav}, {"mop", mop, deltau}, {"ΔGop (kcal/mol)", DeltaGop,
    ErrordeltaG}, {"Keq (op) ", Keqop, ErrorKeqop}, {"meq" , meq, Errormeq},
   {"βT" , βT, ErrorβT}, {"Simulated Chevron Plot", III3CChevronPlot, ""}},
  Frame → All, Alignment → {Left}, Background →
   {{None, None}(*{Gray,White,Gray,White,Gray,White,Gray,White,Gray,White}*)},
  ItemStyle → {{Automatic, Automatic, Automatic}(*{White,Automatic,
      White,Automatic,White,Automatic,White,Automatic,White,Automatic}*)
    (*,{{4,2}→Blue,{5,2}→Red}*)}, Spacings → {0.75, 1}]
```

Parameters at 0M Urea	Value	Error
k_{op} (s^{-1})	0.00189692	0.000024
k_{cl} (s^{-1})	0.549814	0.0101957
Closed state half-life(s)	365.329	4.62217
Open state half-life(s)	1.26043	0.0233732
m_{cl}	-0.779821	0.00957587
m_{op}	1.53215	0.00660216
ΔG_{op} (kcal/mol)	3.40161	0.0134693
$K_{eq\ (op)}$	0.00345012	0.0000774512
meq	-1.38718	0.00611743
β_T	0.337298	0.0000779885
Simulated Chevron Plot		

Experimental simulation & design

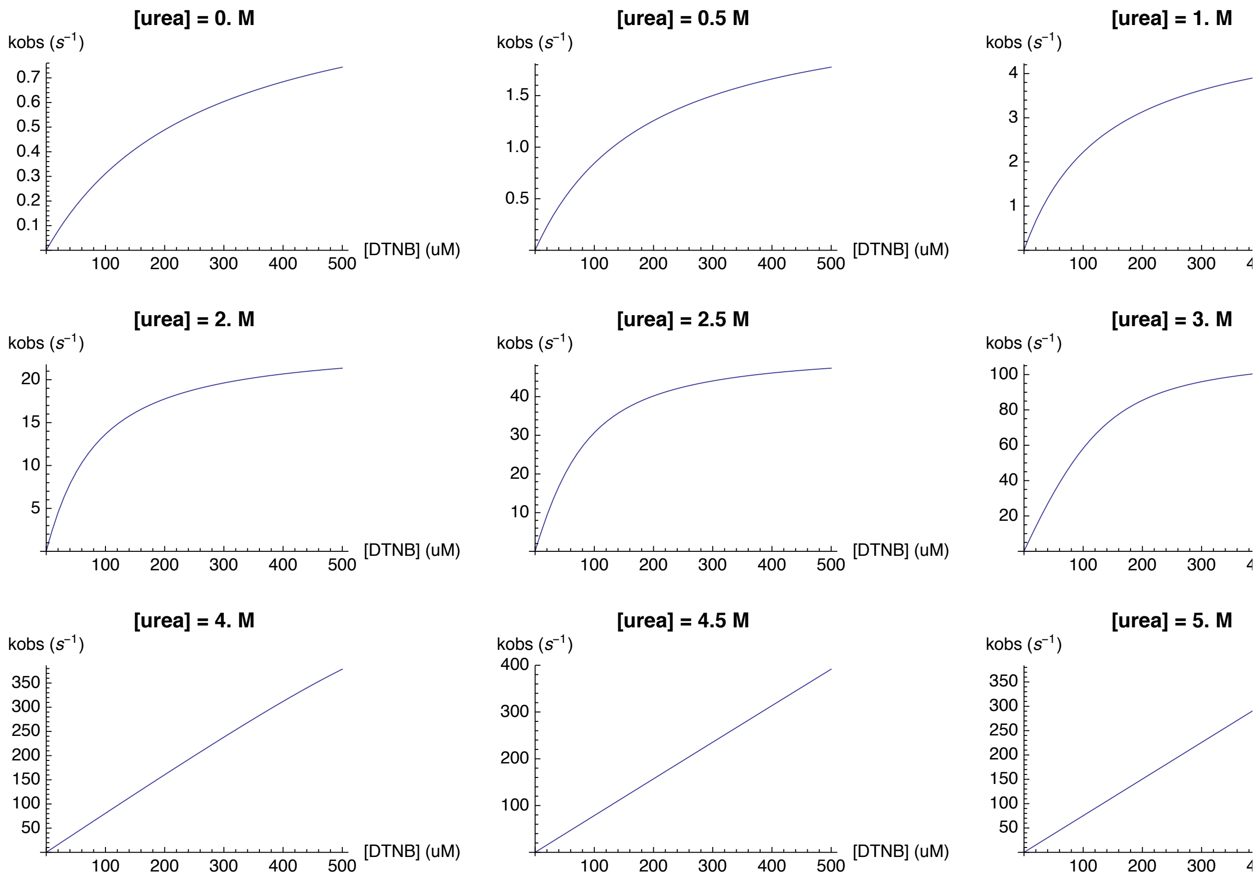
Simulate transients as a function of [urea], with reagent concentration dependent on [urea]

```
Rconc[R_, UreaConc_] := R / 10^UreaConc;
tMax = 1800;
Plot3D[Signal[Rconc[100, UreaConc], UreaConc, t],
 {UreaConc, 0, 3}, {t, 0, tMax}, PlotStyle → Red, PlotRange → Automatic]
```



Simulate kobs vs. [R] curves at various [urea]

```
kobs[Rconc_, Uconc_] := Module[{kop, kcl, kchem},

  kop = Exp[Log[kop0] + mop * Uconc];
  kcl = Exp[Log[kcl0] + mcl * Uconc];
  kchem = Exp[Log[kchem0] - 0.1 * Uconc];
  RateMatrix = ( -kop     kcl                    0
                  kop    -kcl - Rconc * kchem    0
                  0       Rconc * kchem          0 );
  Eigenvalues[RateMatrix]]

kobsPlots = Table[Plot[-kobs[10^-6 Rconc, Uconc][[2]] 600, {Rconc, 0, 500},
    PlotRange → Full, AxesLabel → {"[DTNB]  (uM)", "kobs  (s^-1)"},
    PlotLabel → Style["[urea]  =  " <> ToString[Uconc] <> "  M",
      Bold, FontFamily → "Arial"]], {Uconc, 0, 5.5, 0.5}];
k = 0;
GraphicsGrid[Table[k++;
  kobsPlots[[k]], {i, 3}, {j, i, i + 3}], ImageSize → 1000]
```


[urea] = 0. M
kobs (s^{-1})
[DTNB] (uM)
[urea] = 0.5 M
kobs (s^{-1})
[DTNB] (uM)
[urea] = 1. M
kobs (s^{-1})
[urea] = 2. M
kobs (s^{-1})
[DTNB] (uM)
[urea] = 2.5 M
kobs (s^{-1})
[DTNB] (uM)
[urea] = 3. M
kobs (s^{-1})
[urea] = 4. M
kobs (s^{-1})
[DTNB] (uM)
[urea] = 4.5 M
kobs (s^{-1})
[DTNB] (uM)
[urea] = 5. M
kobs (s^{-1})

Classification of kinetic regimes in diff Urea Conc

```
kop0 = kop0;
kcl0 = kcl0;
mop = mop;
mcl = mcl;
kop = Exp[Log[kop0] + mop * Uconc];
kcl = Exp[Log[kcl0] + mcl * Uconc];
kchem = Exp[Log[kchem0] - 0.1 * Uconc];
Uconc = DeleteDuplicates[Flatten[Join[UreaConc, UreaConcSF1]]];
Rconc = 10^-4;
kint = Rconc * kchem;
Regimes = Table[If[kcl[[i]] ≥ 10 (kint[[i]]), "EX2", If[
      10 (kcl[[i]]) ≤ kint[[i]], "EX1", "EXX", {i, Length[kcl]}]], {i, Length[kcl]}];
Ureaconc = Flatten[{"Urea Conc(M)", Uconc}];
Regime = Flatten[{"Kinetic Regime", Regimes}];
(*EX2=kcl>>kint (10X>>1X)
EX1=kcl <<kint (1X <<10X)
EXX=Diff between kcl& kint<10X
EXX→EX1=EXX close to EX1
EXX→EX2=EXX close to EX2
Beyond E\.18X1:kobs=kint*)
OutputRegimes3C = Grid[{Table[Ureaconc[[i]], {i, Length[Ureaconc]}],
    Table[Regime[[i]], {i, Length[Ureaconc]}]}, Frame → All,
  Alignment → {Left}, Background → {{Gray, LightBlue, LightBlue,
      LightBlue, LightBlue, LightBlue, LightBlue, LightBlue, LightGreen}},
  ItemStyle → {{White}, {Automatic}, {Blue}, {Green}}, Spacings → {1, 1}]
```

Urea Conc(M)	0	0.5	0.75	1	2	3	4	8
Kinetic Regime	EXX	EXX	EXX	EXX	EXX	EXX	EXX	EX1

Exporting data to .pdf format

```
Export["FigforpaperIII3Cfinal.pdf", FigforpaperIII3C];
Export["FigIII3Clegendsfinal.pdf", FigIII3Clegends];
Export["FigIII3CSpecfinalfit.pdf", FigIII3CSpec];
Export["FigIII3CSFfinalfit.pdf", FigIII3CSF];
Export["III3Cfinalparameters.pdf", Output3C];
Export["Forpaper3CAllfinalfit.pdf", FigforpaperIII3CAll];
Export["III3Ckineticregimefinal.pdf", OutputRegimes3C];
```