Basic Setup and Data Input

```
(*Clear all previous definitions and settings*)
ClearAll["Global`*"]
(*Common settings for graphs*)
GR={PlotStyle→Thickness[0.005],
    Frame→True,
    FrameTicks→{{All,All},{All,False}},
    ImageSize→{500,Automatic},
    LabelStyle→{FontFamily→"Times New Roman",FontSize→9},
    FrameLabel→{{"[mg/L]",None},{"Time [d]",None}}};
(*Total number of parameters in the dataset*)
totalParameters = 22;
(*Import the CSV data*)
CPRaw = Import["/.../CP.csv"];
(*Display the header of the table*)
rawDataTable =
  TableForm[CPRaw⟦1⟧⟦1 ;; totalParameters⟧, TableHeadings → {Automatic, None}];
(*Extract the data without headers*)
CP = CPRaw⟦2 ;; -1⟧;
(*Extract time data*)
timeData = Table[sublist⟦1⟧, {sublist, CP}];
(*Display headings of the parameters*)
rawDataTable
```

In[297]:=

1	"Time(d)"
2	"Time(d)"
3	"Inf. NH4-N"
4	"Eff.NH4-N"
5	"Inf.NO2-N"
6	"Eff.NO2-N"
7	"Inf.NO3-N"
8	"Eff. NO3-N"
9	"Inf.COD"
10	"E.sCOD"
11	"E.tCOD"
12	"Inf.pH"
13	"Inf.Alk"
14	"Eff.pH"
15	"Eff.Alk"
16	"Qg"
17	"Eff.TSS"
18	"Eff.VSS"
19	"pCOD/VSS"
20	"HRT"
21	"NLR"
22	"NRR"

Delta Calculations

In[299]:=

```
(*Calculate changes in different parameters*)
deltaNH4 = CP⟦All, 4⟧ - CP⟦All, 3⟧;
deltaNO2 = CP⟦All, 6⟧ - CP⟦All, 5⟧;
deltaNO3 = CP⟦All, 8⟧ - CP⟦All, 7⟧;
deltaAlk = CP⟦All, 15⟧ - CP⟦All, 13⟧;

(*Collect all delta data*)
deltaData = {deltaNH4, deltaNO2, deltaNO3};
```

Alkalinity coupled nitrogen balance

Model Setup

In[304]:=

```
(*Define stoichiometric coefficients for different reactions*)
stoichNH4 = {-1, -1, 0, 0, 0};
stoichNO2 = {-1.32, 1, -1, 1, -1};
stoichNO3 = {0.26, 0, 1, -1, 0};
stoichAlk = {1.74, -7.14, 0, 0, 3.57};
(*Define variable names and styles*)
variableNames = {"ANA", "AOB", "NOB", "DN3", "DN2"};
colors = {Red, Darker[Purple], Darker[Green], Darker[Blue], Orange};
markerStyles = {"■", "●", "●", "▲", "▲"};
(*Create matrices for solving equations*)
xMatrix = Transpose[{ANA, AOB, NOB, DN3, DN2}];
bMatrix = Transpose[{"dNH4", "dNO2", "dNO3"}];
aMatrix = {stoichNH4, stoichNO2, stoichNO3};

(*Display the overall matrix equation*)
MatrixForm[aMatrix].MatrixForm[xMatrix] == MatrixForm[bMatrix]
```

Out[314]=

$$\begin{pmatrix} -1 & -1 & 0 & 0 & 0 \\ -1.32 & 1 & -1 & 1 & -1 \\ 0.26 & 0 & 1 & -1 & 0 \end{pmatrix} . \begin{pmatrix} \mathrm{ANA} \\ \mathrm{AOB} \\ \mathrm{NOB} \\ \mathrm{DN3} \\ \mathrm{DN2} \end{pmatrix} == \begin{pmatrix} \mathrm{dNH4} \\ \mathrm{dNO2} \\ \mathrm{dNO3} \end{pmatrix}$$

Scenario Generation

In[315]:=

```
(*Initialize lists for storing matrices and settings*)
newAMatrixList = {};
newXMatrixList = {};
inverseAMatrixList = {};
inverseARoundedMatrixList = {};
colorList = {};
markerList = {};
varList = {};
(*Generate scenarios by eliminating two activities each time*)
For[i = 2, i ≤ 4, i++, For[j = i + 1, j ≤ 5, j++, newAMatrix = aMatrix[[All, {1, i, j}]];
  newXMatrix = xMatrix[[{1, i, j}]];
  inverseAMatrix = Inverse[newAMatrix];
  inverseAMatrixRounded = SetPrecision[inverseAMatrix, 2];
  newAMatrixList = Append[newAMatrixList, MatrixForm[newAMatrix]];
  newXMatrixList = Append[newXMatrixList, MatrixForm[newXMatrix]];
  inverseARoundedMatrixList =
   Append[inverseARoundedMatrixList, MatrixForm[inverseAMatrixRounded]];
  inverseAMatrixList = Append[inverseAMatrixList, MatrixForm[inverseAMatrix]];
  newVar = variableNames[[{1, i, j}]];
  newColor = colors[[{1, i, j}]];
  newMarker = markerStyles[[{1, i, j}]];
  colorList = Append[colorList, newColor];
  markerList = Append[markerList, newMarker];
  varList = Append[varList, newVar];]]
(*Create grid for displaying scenarios*)
heading = {"Scenario", "A Matrix", "Inverse A Matrix",
   "Picked Activity", "Color picked", "Marker picked"};
grid =
 Grid[{{Item[heading[[1]], Alignment → Center], Item[heading[[2]], Alignment → Center],
    Item[heading[[3]], Alignment → Center], Item[heading[[4]], Alignment → Center]},
   Sequence @@ Table[{i, newAMatrixList[[i]], inverseARoundedMatrixList[[i]],
      MatrixForm[varList[[i]]]}, {i, 1, 6}]}, Dividers → All]
```

Inverse : Result for Inverse of badly conditioned matrix {{−1., 0., 0. }, {−1.32, −1., 1. }, {0.26, 1., −1.}} may contain significant numerical errors.

Out[324]=

Scenario	A Matrix	Inverse A Matrix	Picked Activity
1	$\begin{pmatrix} -1 & -1 & 0 \\ -1.32 & 1 & -1 \\ 0.26 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -0.49 & -0.49 & -0.49 \\ -0.51 & 0.49 & 0.49 \\ 0.13 & 0.13 & 1.1 \end{pmatrix}$	$\begin{pmatrix} ANA \\ AOB \\ NOB \end{pmatrix}$
2	$\begin{pmatrix} -1 & -1 & 0 \\ -1.32 & 1 & 1 \\ 0.26 & 0 & -1 \end{pmatrix}$	$\begin{pmatrix} -0.49 & -0.49 & -0.49 \\ -0.51 & 0.49 & 0.49 \\ -0.13 & -0.13 & -1.1 \end{pmatrix}$	$\begin{pmatrix} ANA \\ AOB \\ DN3 \end{pmatrix}$
3	$\begin{pmatrix} -1 & -1 & 0 \\ -1.32 & 1 & -1 \\ 0.26 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} -6.0\times10^{-17} & -9.3\times10^{-17} & 3.8 \\ -1.0 & -8.2\times10^{-17} & -3.8 \\ -1.0 & -1.0 & -8.9 \end{pmatrix}$	$\begin{pmatrix} ANA \\ AOB \\ DN2 \end{pmatrix}$
4	$\begin{pmatrix} -1 & 0 & 0 \\ -1.32 & -1 & 1 \\ 0.26 & 1 & -1 \end{pmatrix}$	$\begin{pmatrix} -1.0 & 0 & 0 \\ 1.6\times10^{16} & -1.5\times10^{16} & -1.5\times10^{16} \\ 1.6\times10^{16} & -1.5\times10^{16} & -1.5\times10^{16} \end{pmatrix}$	$\begin{pmatrix} ANA \\ NOB \\ DN3 \end{pmatrix}$
5	$\begin{pmatrix} -1 & 0 & 0 \\ -1.32 & -1 & -1 \\ 0.26 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} -1.0 & 0 & 0 \\ 0.26 & -5.5\times10^{-17} & 1.0 \\ 1.1 & -1.0 & -1.0 \end{pmatrix}$	$\begin{pmatrix} ANA \\ NOB \\ DN2 \end{pmatrix}$
6	$\begin{pmatrix} -1 & 0 & 0 \\ -1.32 & 1 & -1 \\ 0.26 & -1 & 0 \end{pmatrix}$	$\begin{pmatrix} -1.0 & 0 & 0 \\ -0.26 & 5.5\times10^{-17} & -1.0 \\ 1.1 & -1.0 & -1.0 \end{pmatrix}$	$\begin{pmatrix} ANA \\ DN3 \\ DN2 \end{pmatrix}$

In[325]:=

Scenario Filtering and Calculation

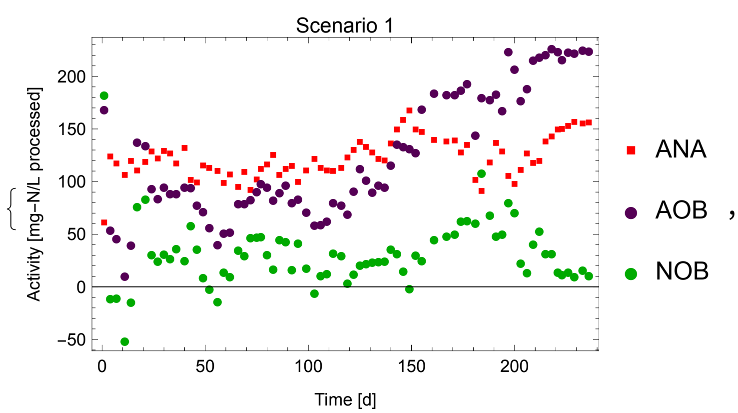
In[326]:=

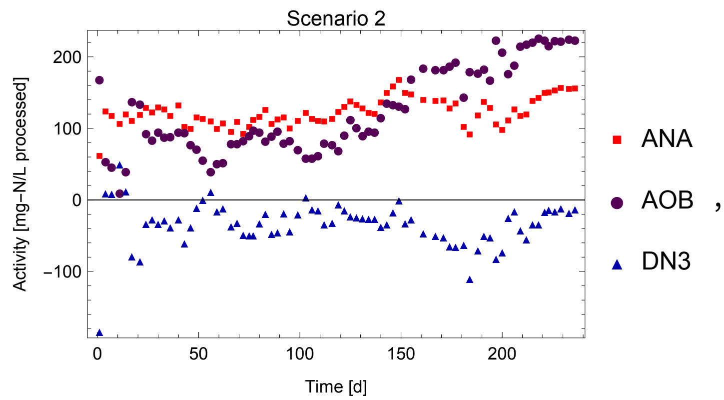
```
(*Remove the ill scenario*)
varList = Delete[varList, 4];
colorList = Delete[colorList, 4];
markerList = Delete[markerList, 4];
inverseAMatrixList = Delete[inverseAMatrixList, 4];
numberOfScenarios = 5;
```

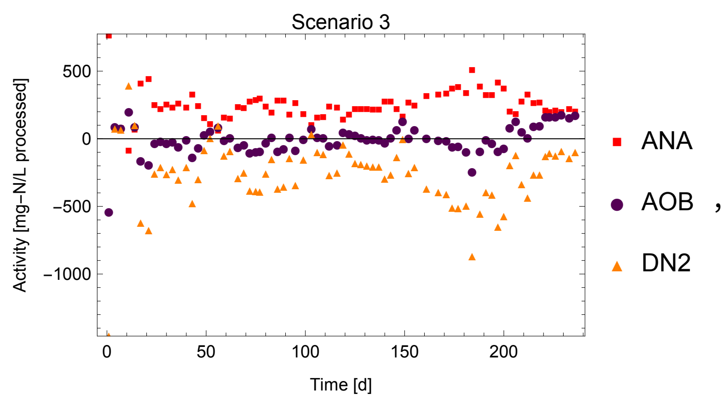
In[331]:=

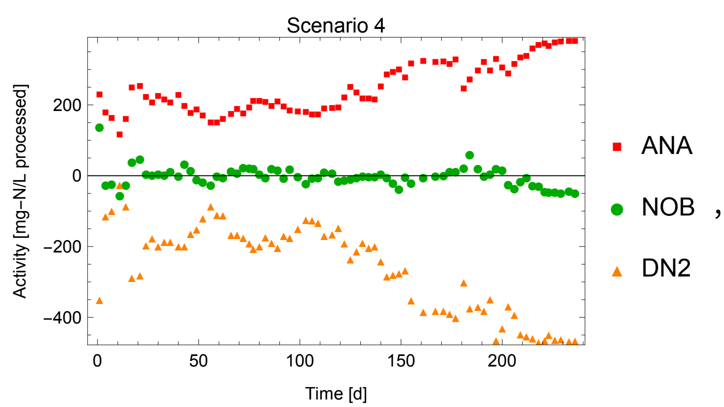
```
(*Calculate the results for each scenario*)
Data = Transpose[{deltaData⟦1⟧, deltaData⟦2⟧, deltaData⟦3⟧}];
ResultList = {};
PlotList = {};
For[i = 1, i ≤ 5, i++,
  Subresult = {};
  For[n = 1, n ≤ Length[timeData], n++,
   Result = inverseAMatrixList⟦i⟧⟦1⟧.Data⟦n⟧;
   Subresult = Append[Subresult, Result];];
  ResultList = Append[ResultList, Subresult];
  plots = {};
  For[j = 1, j ≤ 3, j++, data = #⟦j⟧ & /@ Subresult;
   minValue = Min[Select[Flatten[Subresult], NumericQ]];
   maxValue = Max[Select[Flatten[Subresult], NumericQ]];
   plot = ListPlot[Transpose[{timeData, data}],
     Frame → True,
     FrameLabel → {"Time [d]", "Activity [mg-N/L processed]"},
     PlotLabel → "Scenario " <> ToString[i],
     ImageSize → 300,
     PlotLegends → {varList⟦i⟧⟦j⟧},
     PlotStyle → {colorList⟦i⟧⟦j⟧},
     PlotMarkers → {markerList⟦i⟧⟦j⟧, Scaled[0.03]},
     PlotRange → {Automatic, {minValue - 10, maxValue + 10}}];
   AppendTo[plots, plot];];
  AppendTo[PlotList, Show[plots]];];
PlotList
```

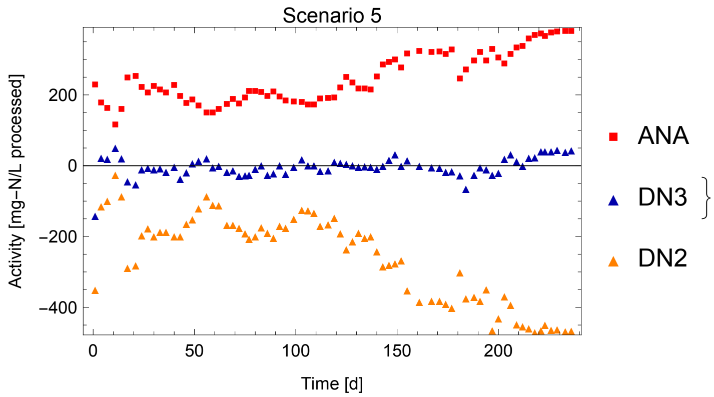
Out[335]=




Scenario 2
Activity [mg-N/L processed]
200
100
0
−100
0
50
100
150
200
Time [d]
ANA
AOB ,
DN3


Scenario 3
Activity [mg-N/L processed]
500
0
−500
−1000
0
50
100
150
200
Time [d]
ANA
AOB ,
DN2


Scenario 4
Activity [mg-N/L processed]
200
0
−200
−400
0
50
100
150
200
Time [d]
ANA
NOB ,
DN2

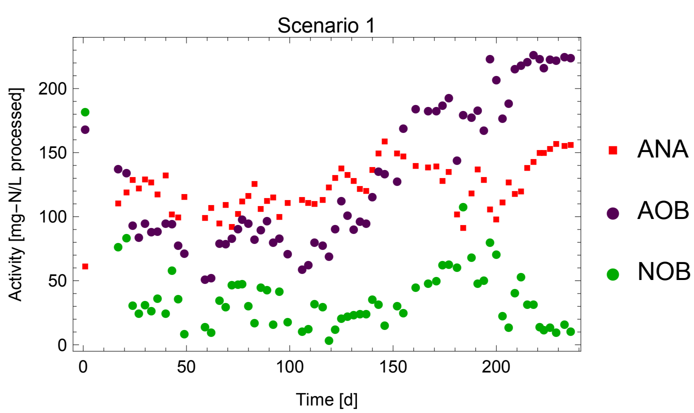


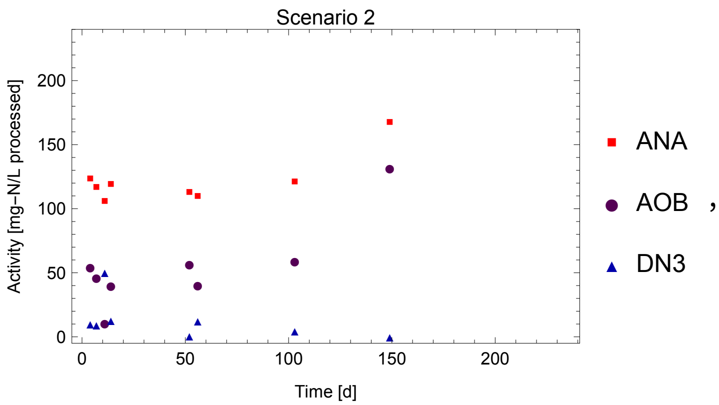
Filtering and Final Data Combination

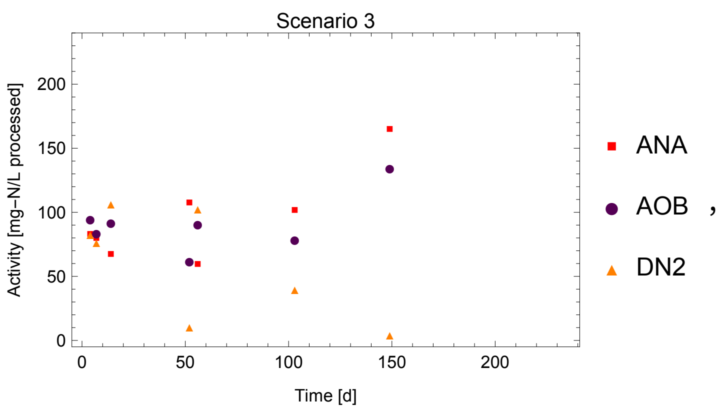
In[336]:=

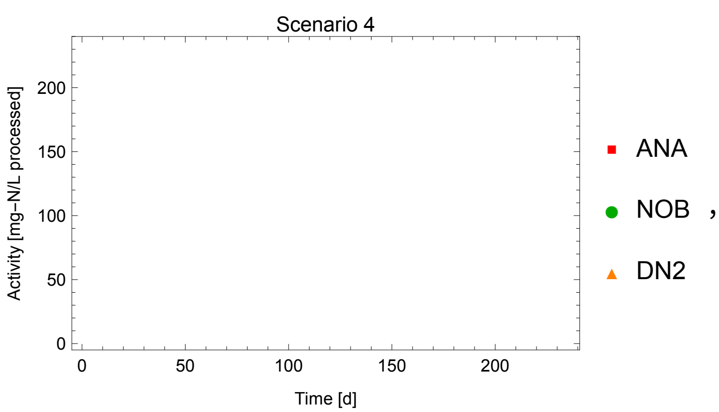
```
(*Filter negative values from the dataset*)
FilteredResultList = {};
For[j = 1, j ≤ 5, j++,
 FilteredSubResult = {};
 For[k = 1, k ≤ Length[Subresult], k++,
  If[Min[ResultList[[j]][[k]]] < 0, ResultList[[j]][[k]] = {-9999, -9999, -9999}];
  FilteredSubResult = Append[FilteredSubResult, ResultList[[j]][[k]]]];
 FilteredResultList = Append[FilteredResultList, FilteredSubResult];
 ];
(*Generate plots for filtered data*)
PlotList2 = {};
TotalFilterPlot = {};
For[i = 1, i ≤ 5, i++,
 plots = {};
 For[j = 1, j ≤ 3, j++,
  maxValue = Max[Select[Flatten[FilteredResultList[[i]]], NumericQ]];
  If[maxValue == -9999, maxValue = 100];
  plot = ListPlot[Transpose[{timeData, FilteredResultList[[i]][[All, j]]}],
   Frame → True,
   FrameLabel → {"Time [d]", "Activity [mg-N/L processed]"},
   PlotLabel → "Scenario " <> ToString[i],
   ImageSize → 300,
   PlotStyle → {colorList[[i]][[j]]},
   PlotMarkers → {markerList[[i]][[j]], Scaled[0.03]},
   PlotRange → {{Automatic, Automatic}, {-5, 240}},
   PlotLegends → {varList[[i]][[j]]}];
  AppendTo[plots, plot];];
 AppendTo[PlotList2, Show[plots]];];
AppendTo[TotalFilterPlot, PlotList2];
TotalFilterPlot
```

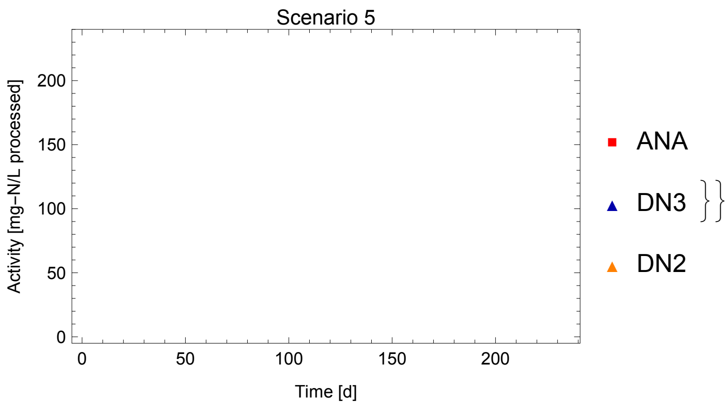
Out[342]=

{{  ,


Scenario 2
Activity [mg-N/L processed]
Time [d]
0
50
100
150
200
ANA
AOB
DN3
,


Scenario 3
Activity [mg-N/L processed]
Time [d]
0
50
100
150
200
ANA
AOB
DN2
,


Scenario 4
Activity [mg-N/L processed]
Time [d]
0
50
100
150
200
ANA
NOB
DN2
,



In[343]:=

```
(*Extract the positive results for each scenario*)
dataset1 = FilteredResultList[[1]];
dataset2 = FilteredResultList[[2]];
dataset3 = FilteredResultList[[3]];
dataset4 = FilteredResultList[[4]];
dataset5 = FilteredResultList[[5]];

(*Calculate the alkalinity corresponding to each scenario:alk1,
alk2,alk3,alk4,alk5*)
alk1 = dataset1[[All, 1]] * 1.74 - 7.14 * dataset1[[All, 2]];
alk2 = dataset2[[All, 1]] * 1.74 - 7.14 * dataset2[[All, 2]];
alk3 = dataset3[[All, 1]] * 1.74 - 7.14 * dataset3[[All, 2]] + 3.57 * dataset3[[All, 3]];
alk4 = dataset4[[All, 1]] * 1.74 + 3.57 * dataset4[[All, 3]];
alk5 = dataset5[[All, 1]] * 1.74 + 3.57 * dataset5[[All, 3]];
(*Pick the dataset that converge the best with measured alkalinity*)
errors1 = Abs[Part[alk1, #] - deltaAlk[[#]]] & /@ Range[Length[deltaAlk]];
errors2 = Abs[Part[alk2, #] - deltaAlk[[#]]] & /@ Range[Length[deltaAlk]];
errors3 = Abs[Part[alk3, #] - deltaAlk[[#]]] & /@ Range[Length[deltaAlk]];
errors4 = Abs[Part[alk4, #] - deltaAlk[[#]]] & /@ Range[Length[deltaAlk]];
errors5 = Abs[Part[alk5, #] - deltaAlk[[#]]] & /@ Range[Length[deltaAlk]];
minErrors = MapThread[Min, {errors1, errors2, errors3, errors4, errors5}];
bestScenarios =
  Table[Which[errors1[[i]] == minErrors[[i]], Append[Part[dataset1[[i]]], "S1"],
    errors2[[i]] == minErrors[[i]], Append[Part[dataset2[[i]]], "S2"],
    errors3[[i]] == minErrors[[i]], Append[Part[dataset3[[i]]], "S3"],
    errors4[[i]] == minErrors[[i]], Append[Part[dataset4[[i]]], "S4"],
    errors5[[i]] == minErrors[[i]], Append[Part[dataset5[[i]]], "S5"]],
   {i, Length[deltaAlk]}];
(*Update the uncalculated activties to -9999 based on the selected scenario*)
bestScenarios = Map[Function[data,
    Switch[data[[-1]], "S1", Insert[Insert[data, -9999, 4], -9999, 5], "S2",
```

```
        Insert[Insert[data, -9999, 3], -9999, 5], "S3", Insert[Insert[data, -9999, 3],
         -9999, 4], "S4", Insert[Insert[data, -9999, 2], -9999, 4], "S5",
        Insert[Insert[data, -9999, 2], -9999, 3], _, data]], bestScenarios];
(*Data visualization*)
subset = bestScenarios;
plots = {};
For[i = 1, i ≤ 5, i++,
 plot = ListPlot[Transpose[{timeData, bestScenarios⟦All, i⟧}],
   Frame → True,
   FrameLabel → {"Time [d]", "Activity [mg-N/L processed]"},
   ImageSize → 300,
   PlotLegends → {variableNames⟦i⟧},
   PlotStyle → {colors⟦i⟧},
   PlotRange → {{Automatic, Automatic}, {-10, 240}},
   PlotMarkers → {markerStyles⟦i⟧, Scaled[0.03]}];
 AppendTo[plots, plot];]
Show[plots]
```

Out[364]=