

```

In[ ]:= SetDirectory[
    "C:/Users/serha/OneDrive/Masaüstü/MyRepo/master_thesis_MMT003/210519_time_windows_and
    _OR_model/deleting_reactions"];

In[ ]:= Get["../../algorithm_packages/SingleNetworks-algorithm-package-2.wl"]
    (* ?SingleNetworks` * *)

In[ ]:= stoichioforhomosapiens =
    Drop[Import["../../210324_disc_time_windows_and_OR_model/iAT_PLT_636_stoichiomat.csv",
        HeaderLines -> 1], None, {1}];
    SparseArray@stoichioforhomosapiens

Out[ ]:= SparseArray[  Specified elements: 4006
    Dimensions: {738, 1008} ]

In[ ]:= stoichiometricmatrix = stoichioforhomosapiens;
    metabolites = 738;
    fluxexchanges = 1008;
    steadystatevector = ConstantArray[{0, 0}, metabolites];
    first[a_] := First /@ GatherBy[Ordering@a, a[[#]] &] // Sort;

In[ ]:= subsetpositionsforsequences = Import["../cases/subsetpositionsforsequences.mx"];
    boundaries = Import["../cases/boundaries_-5and5_105.mx"];
    boundariespos = Flatten@Position[boundaries, {-5, 5}];
    objfunctions = Import[
        "C:/Users/serha/NonDrive/OR_model-objective_functions/+2+4objfunc_fxdbounds_-5and5_
        105pcs.mx"];

In[ ]:= boundariesa = ReplacePart[ConstantArray[{-500, 500}, fluxexchanges], MapThread[
    #1 -> #2 &, {boundariespos, ConstantArray[{-0.5, 0.5}, Length@boundariespos]}]];

In[ ]:= syntheticseqgenerator[stoichiometricmatrix_, steadystatevector_, boundaries_,
    fluxexchanges_, subsetpositions_, objectivefunctions_] := Module[{solutionvectors},
    solutionvectors = Chop[Table[LinearProgramming[-objectivefunctions[[i]],
        stoichiometricmatrix, steadystatevector, boundaries],
        {i, Length@objectivefunctions}], 10^-5]]

In[ ]:= AbsoluteTiming[solutionvectors = Quiet@Table[syntheticseqgenerator[stoichiometricmatrix,
    steadystatevector, boundariesa, fluxexchanges, i[[1]], i[[2]]],
    {i, MapThread[{#1, #2} &, {subsetpositionsforsequences, objfunctions}}]];

Out[ ]:= {5265.71, Null}

In[ ]:= Export[
    "C:/Users/serha/NonDrive/OR_model-solution_vectors/+2+4solutionvectors_fxdcoeffs_-05
    and05_105pcs.mx", solutionvectors]

Out[ ]:= C:/Users/serha/NonDrive/OR_model-solution_vectors/+2+4solutionvectors_fxdcoeffs_-05
    and05_105pcs.mx

```

```

(*solutionvectors=Import[
  "C:/Users/serha/NonDrive/OR_model-solution_vectors/+2+4solutionvectors_fxdcoeffs_-05
  and05_210pcs.mx"];*)

In[ ]:= SeedRandom@25;
randomreactionlist =
  Table[Sort@RandomInteger[{1, fluxexchanges}, i], {i, Range[1008, 500, -50]}];

In[ ]:= solutionvectorsreactionsdeleted =
  Table[Partition[Flatten[solutionvectors, 1][[All, i]], 300], {i, randomreactionlist}];

In[ ]:= objfunctionsreactionsdeleted =
  Table[Partition[Flatten[objfunctions, 1][[All, i]], 300], {i, randomreactionlist}];

In[ ]:= AbsoluteTiming[
  featuredatalist = Table[Table[MapThread[Dot, {objfunctionsreactionsdeleted[[j]][[i]],
    solutionvectorsreactionsdeleted[[j]][[i]]}], {i, 200}], {j, 10}];]

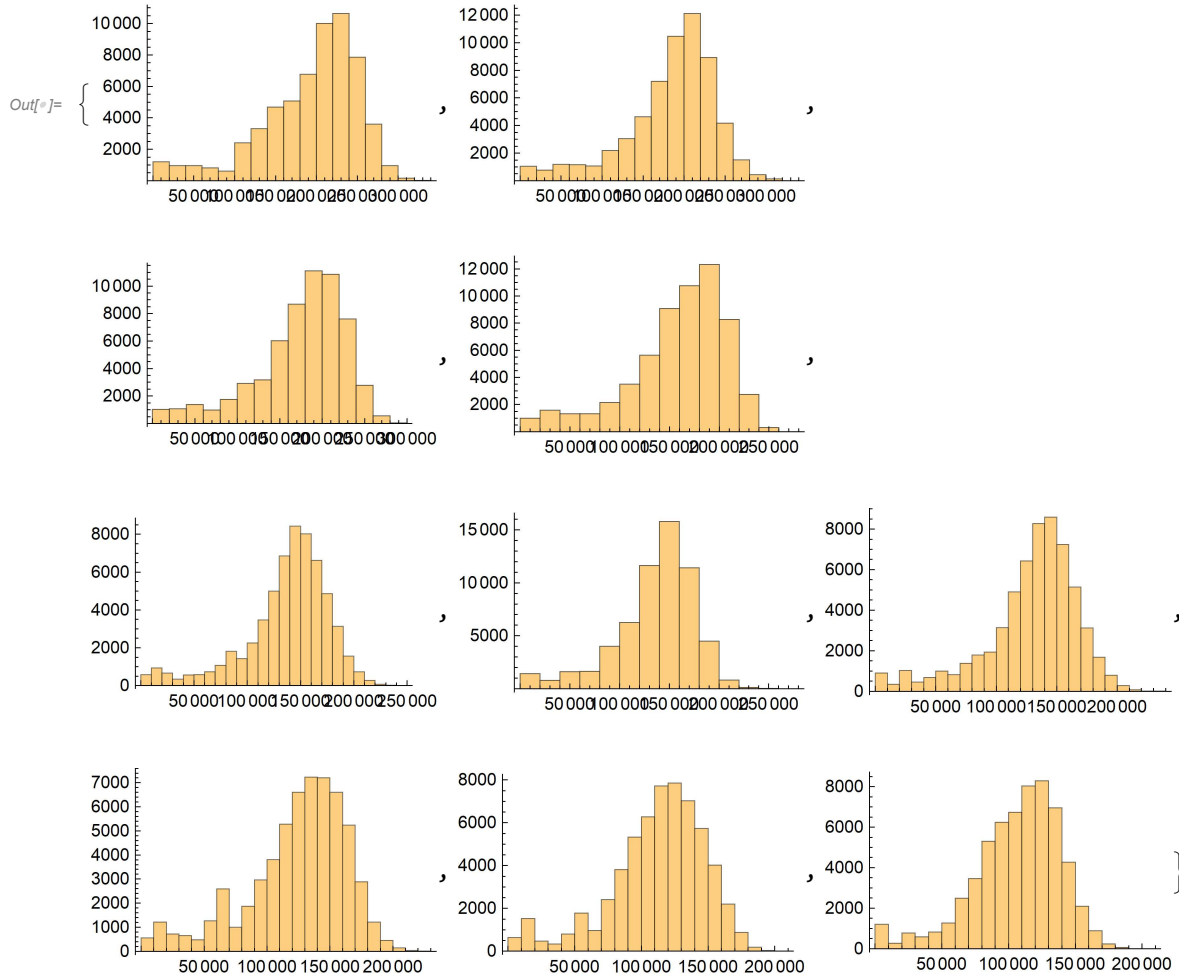
Out[ ]:= {8.41755, Null}

```

```

In[ ]:= datafulllist = Table[Join[Partition[Range@60000, 1],
  Partition[Flatten@Table[ConstantArray[i, 300], {i, 200}], 1],
  Partition[Flatten[featuredatalist[[j]], 1], 1], 2], {j, 10}];
Table[Histogram@datafulllist[[i]][[All, 3]], {i, 10}]

```



```

In[ ]:= thread = Thread[{Range@10, Range[4500, 3150, -150]}]

```

```

Out[ ]:= {{1, 4500}, {2, 4350}, {3, 4200}, {4, 4050},
  {5, 3900}, {6, 3750}, {7, 3600}, {8, 3450}, {9, 3300}, {10, 3150}}

```

```

In[ ]:= AbsoluteTiming[widthdataFixedstep2 =
  Table[snetworkdatabin[3, i[[2]], datafulllist[[i[[1]]]], {i, thread}];]

```

```

Out[ ]:= {72.2758, Null}

```

```

In[ ]:= graphsandnodenumbers12 = Table[snetworkgraph[widthdataFixedstep2[[i]][[1]],
  widthdataFixedstep2[[i]][[2]], 2, 7, 400, Green], {i, 10}];
graphsandnodenumbers12[[All, 2]]

```

```

Out[ ]:= {73, 73, 71, 66, 63, 70, 65, 66, 61, 62}

```

```

In[ ]:= modularityvalues12 = Table[N@GraphAssortativity[graphsandnodenumbers12[[i]][[1]],
    FindGraphCommunities[graphsandnodenumbers12[[i]][[1]]], "Normalized" -> False],
    {i, Length@graphsandnodenumbers12}];

In[ ]:= singlerandomgraphsdegfxd12 =
    Table[randomizinggraphdegfxd[i], {i, graphsandnodenumbers12[[All, 1]]}];
singlerandomerdrenmodularityvalues12 =
    Table[N@GraphAssortativity[singlerandomgraphsdegfxd12[[i]],
        FindGraphCommunities[singlerandomgraphsdegfxd12[[i]]], "Normalized" -> False],
        {i, Length@singlerandomgraphsdegfxd12}];
singlerandomgraphscomm12 = Table[randomizinggraphmod[i],
    {i, graphsandnodenumbers12[[All, 1]]}];
singlerandomcommmodularityvalues12 =
    Table[N@GraphAssortativity[singlerandomgraphscomm12[[i]],
        FindGraphCommunities[singlerandomgraphscomm12[[i]]], "Normalized" -> False],
        {i, Length@singlerandomgraphscomm12}];

In[ ]:= AbsoluteTiming[Zscoresmodularity12 =
    Table[zscorefunctionfortwonullmodels[i], {i, graphsandnodenumbers12[[All, 1]]}];]

Out[ ]:= {210.056, Null}

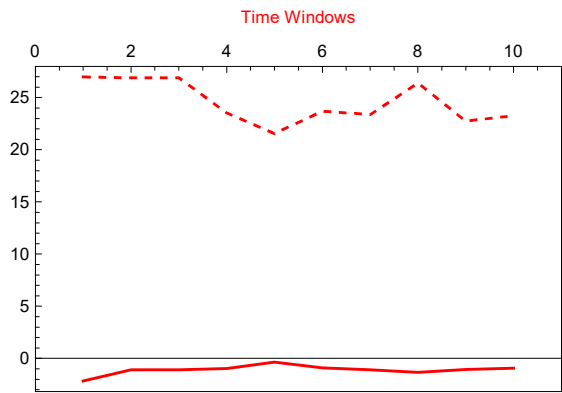
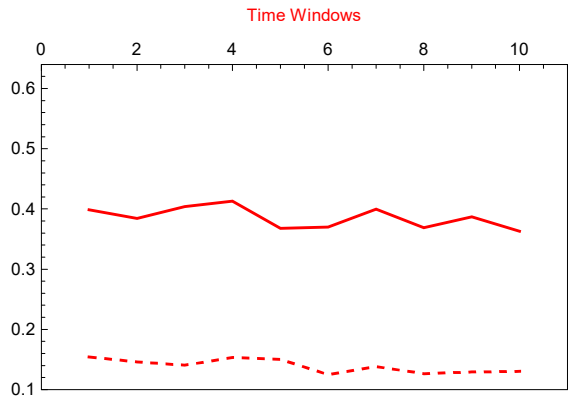
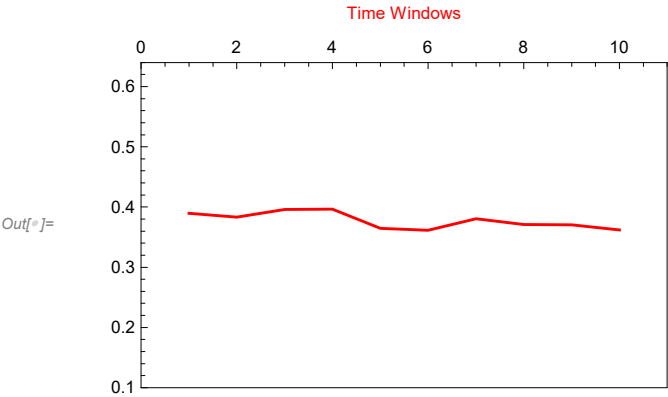
In[ ]:= bucketnode12 = graphsandnodenumbers12[[All, 2]]
Out[ ]:= {73, 73, 71, 66, 63, 70, 65, 66, 61, 62}

```

```

In[ ]:= modularityvaluestimewinsmall = modularityvalues12;
randommodtimewinsmalldegreefxd = singlerandomerdrenmodularityvalues12;
randommodtimewinsmallcomm = singlerandomcommmodularityvalues12;
Zscoretimewinsmall = Zscoresmodularity12;
modularityplotrange = {0.1, 0.64};
(*MinMax[{modularityvalues1,singlerandomcommmodularityvalues1,
singlerandomerdrenmodularityvalues1,modularityvalues12}]*
padding = 38;
win2 = 10;
Row[{ListLinePlot[Thread[{Range@win2, modularityvaluestimewinsmall}],
Frame → True, ImagePadding → padding, FrameTicks → {{All, None}, {None, All}},
FrameLabel → {{None, None}, {None, Style["Time Windows", Red]}}, PlotStyle → Red,
ImageSize → 350, PlotRange → {{0, win2 + 1}, modularityplotrange}],
Row[{ListLinePlot[{Thread[{Range@win2, randommodtimewinsmalldegreefxd}],
Thread[{Range@win2, randommodtimewinsmallcomm}]}], Frame → True,
ImagePadding → padding, FrameTicks → {{All, None}, {None, All}},
FrameLabel → {{None, None}, {None, Style["Time Windows", Red]}},
PlotStyle → {{Dashed, Red}, Red}, ImageSize → 350,
PlotRange → {{0, win2 + 1}, modularityplotrange}],
ListLinePlot[{Thread[{Range@win2, Zscoretimewinsmall[[All, 1]]}],
Thread[{Range@win2, Zscoretimewinsmall[[All, 2]]}]}], Frame → True,
ImagePadding → padding, FrameTicks → {{All, None}, {None, All}},
FrameLabel → {{None, None}, {None, Style["Time Windows", Red]}},
PlotStyle → {{Dashed, Red}, Red}, ImageSize → 350,
PlotRange → {{0, win2 + 1}, MinMax[Flatten[Zscoretimewinsmall], 1]}]}],
LineLegend[{Dashed, Black}, {"Degrees Fixed N.M.", "Modularity N.M."},
LegendMargins → 0, LegendMarkerSize → {20, 20}], Spacer@0.1]]

```



--- Degrees Fixed N.M.
— Modularity N.M.

```
In[ ]:= AbsoluteTiming[widthdataFixedbucket2 =  
  Table[snetworkdatafxdbucket[3, bucketnode12[[i]], datafulllist[[i]]], {i, 10}];]  
Out[ ]:= {18.5782, Null}
```

```

In[ ]:= graphsandnodenumbers32 = Table[snetworkgraph[widthdataFixedbucket2[[i]][[1]],
      widthdataFixedbucket2[[i]][[2]], 1.5, 7, 400, Green], {i, 10}];
modularityvalues32 = Table[N@GraphAssortativity[graphsandnodenumbers32[[i]][[1]],
      FindGraphCommunities[graphsandnodenumbers32[[i]][[1]], "Normalized" -> False],
      {i, Length@graphsandnodenumbers32}];

In[ ]:= singlerandomgraphsdegfxd32 =
      Table[randomizinggraphdegfxd[i], {i, graphsandnodenumbers32[[All, 1]]}];
singlerandomerdrenmodularityvalues32 =
      Table[N@GraphAssortativity[singlerandomgraphsdegfxd32[[i]],
      FindGraphCommunities[singlerandomgraphsdegfxd32[[i]], "Normalized" -> False],
      {i, Length@singlerandomgraphsdegfxd32}];
singlerandomgraphscomm32 = Table[randomizinggraphmod[i],
      {i, graphsandnodenumbers32[[All, 1]]}];
singlerandomcommmodularityvalues32 =
      Table[N@GraphAssortativity[singlerandomgraphscomm32[[i]],
      FindGraphCommunities[singlerandomgraphscomm32[[i]], "Normalized" -> False],
      {i, Length@singlerandomgraphscomm32}];

In[ ]:= AbsoluteTiming[Zscoresmodularity32 =
      Table[zscorefunctionfortwonullmodels[i], {i, graphsandnodenumbers32[[All, 1]]}];]

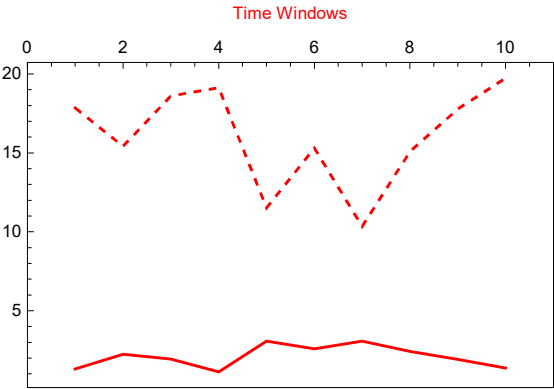
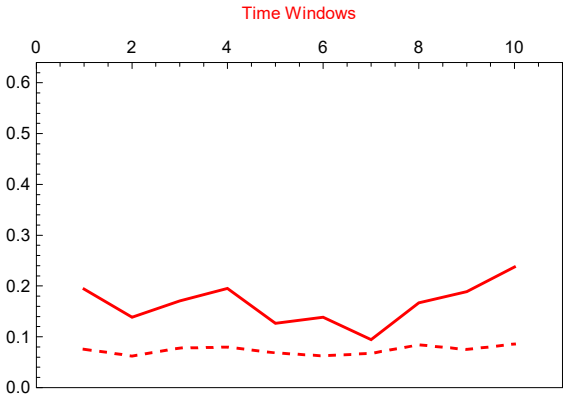
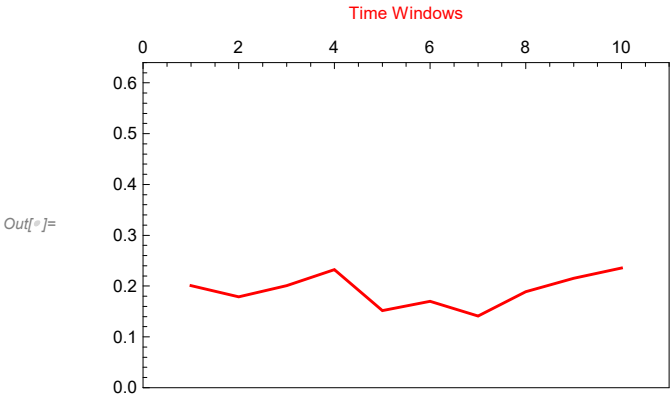
Out[ ]:= {244.659, Null}

```

```

In[ ]:= modularityvaluestimewinsmall = modularityvalues32;
randommodtimewinsmalldegrefxd = singlerandomerdrenmodularityvalues32;
randommodtimewinsmallcomm = singlerandomcommmodularityvalues32;
Zscoretimewinsmall = Zscoresmodularity32;
modularityplotrange = {0, 0.64};
(*MinMax[{modularityvalues1,singlerandomcommmodularityvalues1,
singlerandomerdrenmodularityvalues1,modularityvalues12}]*
padding = 38;
win2 = 10;
Row[{ListLinePlot[Thread[{Range@win2, modularityvaluestimewinsmall}],
Frame → True, ImagePadding → padding, FrameTicks → {{All, None}, {None, All}},
FrameLabel → {{None, None}, {None, Style["Time Windows", Red]}}, PlotStyle → Red,
ImageSize → 350, PlotRange → {{0, win2 + 1}, modularityplotrange}],
Row[{ListLinePlot[{Thread[{Range@win2, randommodtimewinsmalldegrefxd}],
Thread[{Range@win2, randommodtimewinsmallcomm}]}], Frame → True,
ImagePadding → padding, FrameTicks → {{All, None}, {None, All}},
FrameLabel → {{None, None}, {None, Style["Time Windows", Red]}},
PlotStyle → {{Dashed, Red}, Red}, ImageSize → 350,
PlotRange → {{0, win2 + 1}, modularityplotrange}],
ListLinePlot[{Thread[{Range@win2, Zscoretimewinsmall[[All, 1]]}],
Thread[{Range@win2, Zscoretimewinsmall[[All, 2]]}]}], Frame → True,
ImagePadding → padding, FrameTicks → {{All, None}, {None, All}},
FrameLabel → {{None, None}, {None, Style["Time Windows", Red]}},
PlotStyle → {{Dashed, Red}, Red}, ImageSize → 350,
PlotRange → {{0, win2 + 1}, MinMax[Flatten[Zscoretimewinsmall], 1]}]}],
LineLegend[{Dashed, Black}, {"Degrees Fixed N.M.", "Modularity N.M."},
LegendMargins → 0, LegendMarkerSize → {20, 20}], Spacer@0.1]]

```

--- Degrees Fixed N.M.
— Modularity N.M.

```

In[ ]:= Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-modularityvalues-fss.mx",
  modularityvalues12]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-erd-modularityvalues-fss.mx",
  singlerandomerdrenmodularityvalues12]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-comm-modularityvalues-fss.mx",
  singlerandomcommmodularityvalues12]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-zscores-fss.mx", Zscoresmodularity12]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-modularityvalues-fbs.mx",
  modularityvalues32]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-erd-modularityvalues-fbs.mx",
  singlerandomerdrenmodularityvalues32]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-comm-modularityvalues-fbs.mx",
  singlerandomcommmodularityvalues32]
Export["plot_values/fxd_coeffs/-05+05_105_(2,4)-zscores-fbs.mx", Zscoresmodularity32]

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-modularityvalues-fss.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-erd-modularityvalues-fss.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-comm-modularityvalues-fss.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-zscores-fss.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-modularityvalues-fbs.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-erd-modularityvalues-fbs.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-singrand-comm-modularityvalues-fbs.mx

Out[ ]:= plot_values/fxd_coeffs/-05+05_105_(2,4)-zscores-fbs.mx

```