

```

In[ ]:= SetDirectory[
    "C:/Users/serha/OneDrive/Masaüstü/MyRepo/master_thesis_MMT003/210714_finalising/
    fxd_bounds"];

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[ ]:= case = "bounds";
    intvalues = {-1, 1};
    interval2 = "-250+250_105";

    interval = "25percentdecreased_("<>
        ToString@intvalues[[1]] <> ", " <> ToString@intvalues[[2]] <> ")";
    subsetpositionsforsequences = Import[
        "../../cases/subsetpositionsforsequences_25percentdecreased.mx"];
    boundaries = Import["../../cases/boundaries_for_deleted_reaction_series_-5and5_105.mx"];
    boundariespos0 = Table[Position[boundaries[[i]], {0, 0}], {i, 10}];
    boundariesposval = Table[Position[boundaries[[i]], {-5, 5}], {i, 10}];
    boundariesa = Table[ReplacePart[
        (Table[ReplacePart[ConstantArray[{-500, 500}, fluxexchanges], MapThread[#1 → #2 &,
            {boundariespos0[[i]], ConstantArray[{0, 0}, Length@boundariespos0[[i]]}]]],
            {i, 10}][[j]], MapThread[#1 → #2 &, {boundariesposval[[j]],
            ConstantArray[{-250, 250}, Length@boundariesposval[[j]]}]]], {j, 10}];

In[ ]:= syntheticseqgenerator[stoichiometricmatrix_,
    steadystatevector_, boundaries_, fluxexchanges_, subsetpositions_] :=
    Module[{coefficients, objectivefunctions, solutionvectors},
        coefficients = Table[RandomReal[intvalues, Length@subsetpositions], 50];
        objectivefunctions = Table[ReplacePart[ConstantArray[0., fluxexchanges],
            MapThread[#1 → #2 &, {subsetpositions, coefficients[[i]]}]], {i, 50}];
        solutionvectors = Chop[Table[LinearProgramming[-objectivefunctions[[i]],
            stoichiometricmatrix, steadystatevector, boundaries],
            {i, Length@objectivefunctions}], 10^-5];
        {objectivefunctions, solutionvectors}]

```

```

In[6]:= (*AbsoluteTiming[resultset=
      Table[Quiet@Table[syntheticseqgenerator[stoichiometricmatrix,steadystatevector,
        j,fluxexchanges,i],{i,subsetpositionsforsequences}],{j,boundariesa}];]*)

In[6]:= (*Export["C:/Users/serha/NonDrive/OR_model-25.06.2021/solution_vectors/"<>
      interval<>"solutionvectors_fxd"<>case<>"_-250and250_105.mx",
      Table[Flatten[resultset[[i]][[All,2]],1],{i,10}]]

      Export["C:/Users/serha/NonDrive/OR_model-25.06.2021/objective_functions/"<>
        interval<>"objfunc_fxd"<>case<>"_-250and250_105.mx",
        Table[Flatten[resultset[[i]][[All,1]],1],{i,10}]]*)

In[6]:= (*solutionvectorslist=Table[Flatten[resultset[[i]][[All,2]],1],{i,10}];
      objfunctionslist=Table[Flatten[resultset[[i]][[All,1]],1],{i,10}];*)

In[6]:= solutionvectorslist =
      Import["C:/Users/serha/NonDrive/OR_model-25.06.2021/solution_vectors/"<>
        interval<>"solutionvectors_fxd"<>case<>"_-250and250_105.mx"];
      objfunctionslist = Import[
        "C:/Users/serha/NonDrive/OR_model-25.06.2021/objective_functions/"<>
        interval<>"objfunc_fxd"<>case<>"_-250and250_105.mx"];

In[6]:= AbsoluteTiming[featuredatalist =
      Table[MapThread[Dot, {objfunctionslist[[j]], solutionvectorslist[[j]]}], {j, 10}];]

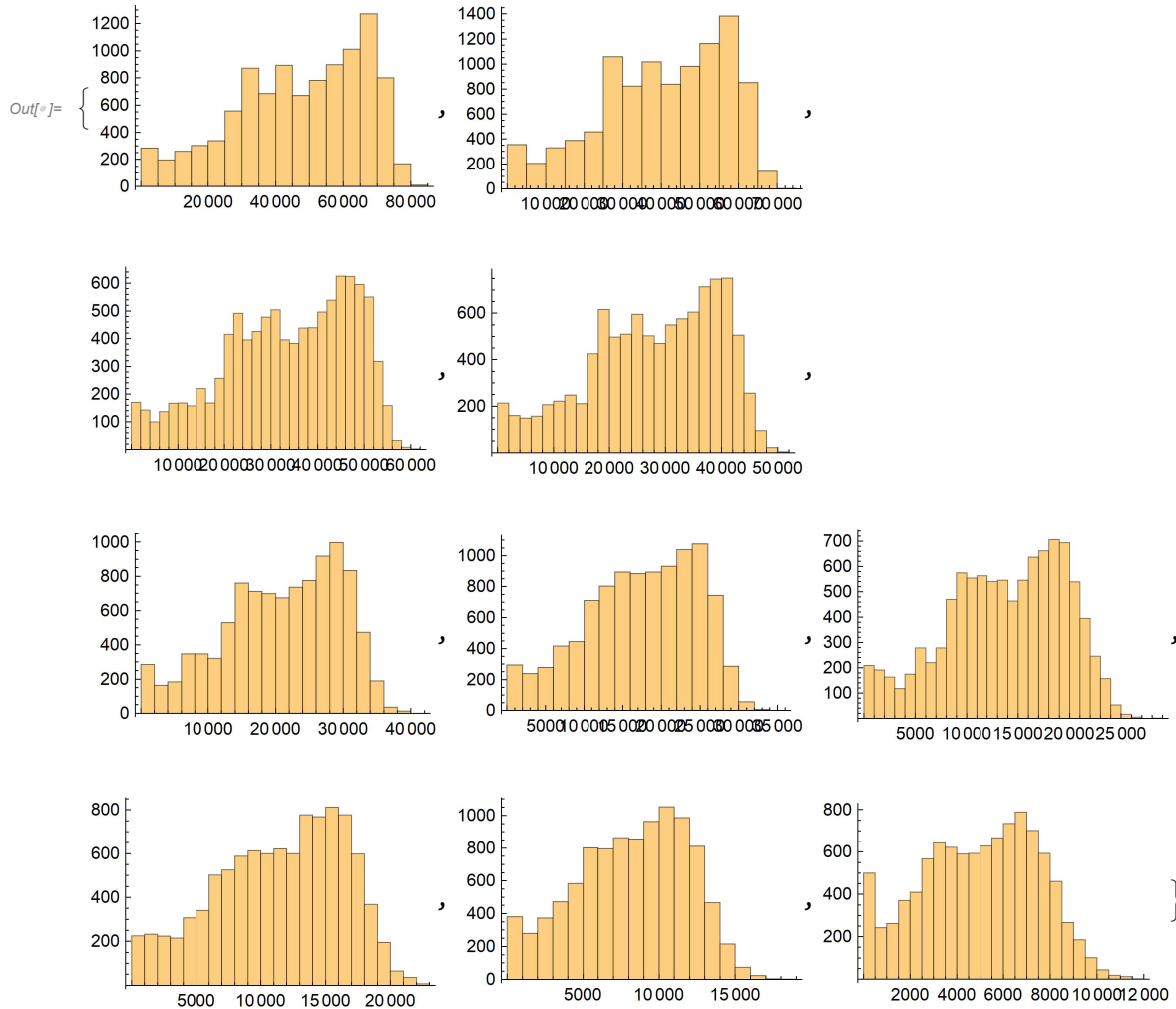
Out[6]:= {2.6943, Null}

```

```

In[ ]:= datafulllist = Table[Join[Partition[Range@10000, 1],
  Partition[Flatten@Table[ConstantArray[i, 50], {i, 200}], 1],
  Partition[featuredatalist[[j]], 1], 2], {j, 10}];
Table[Histogram@datafulllist[[i]][[All, 3]], {i, 10}]

```



```

In[ ]:= thread = {{1, 1950}, {2, 1700}, {3, 1400}, {4, 1200},
  {5, 900}, {6, 800}, {7, 600}, {8, 500}, {9, 440}, {10, 250}};
Mean@thread[[All, 2]]

```

Out[]:= 974

```

In[ ]:= thread = Thread[{Range@10, 2965}]

```

```

Out[ ]:= {{1, 2965}, {2, 2965}, {3, 2965}, {4, 2965},
  {5, 2965}, {6, 2965}, {7, 2965}, {8, 2965}, {9, 2965}, {10, 2965}}

```

```

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

```

Out[]:= {3.85956, Null}

```

In[ ]:= graphsandnodenumbers12 = Table[NetworkGraph[widthdataFixedstep2[[i]][[1]],
    widthdataFixedstep2[[i]][[2]], 2, 7, 400, Green], {i, 10}];
graphsandnodenumbers12[[All, 2]]
Out[ ]:= {28, 25, 21, 18, 14, 12, 10, 8, 7, 4}

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]]}];
singerandomerdrenmodularityvalues12 =
    Table[N@GraphAssortativity[singlerandomgraphsdegfxd12[[i]],
        FindGraphCommunities[singlerandomgraphsdegfxd12[[i]], "Normalized" -> False],
        {i, Length@singlerandomgraphsdegfxd12}];
singerandomgraphscomm12 = Table[randomizinggraphmod[i],
    {i, graphsandnodenumbers12[[All, 1]]}];
singerandomcommmodularityvalues12 =
    Table[N@GraphAssortativity[singlerandomgraphscomm12[[i]],
        FindGraphCommunities[singlerandomgraphscomm12[[i]], "Normalized" -> False],
        {i, Length@singlerandomgraphscomm12}];

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

... Power: Infinite expression  $\frac{1}{0}$  encountered.
... Infinity: Indeterminate expression 0. ComplexInfinity encountered.
... Power: Infinite expression  $\frac{1}{0}$  encountered.
... Infinity: Indeterminate expression 0. ComplexInfinity encountered.

Out[ ]:= {62.0252, Null}


In[ ]:= bucketnode12 = graphsandnodenumbers12[[All, 2]]
Out[ ]:= {28, 25, 21, 18, 14, 12, 10, 8, 7, 4}

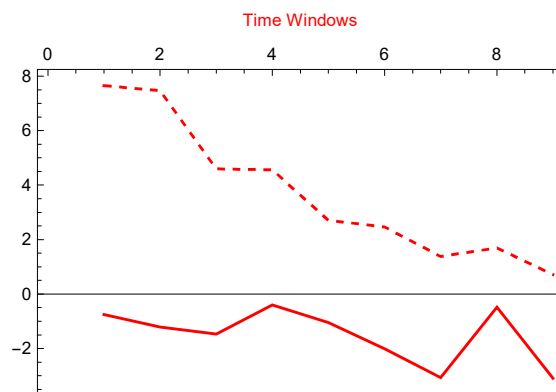
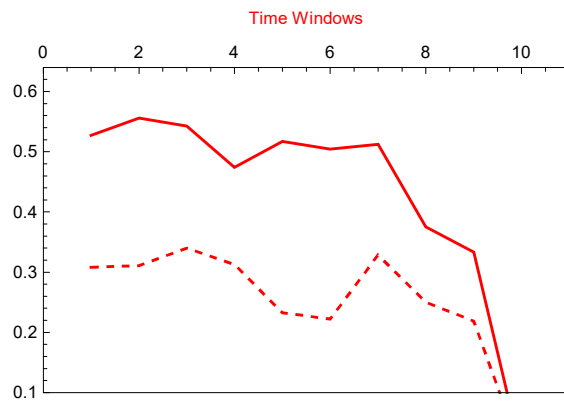
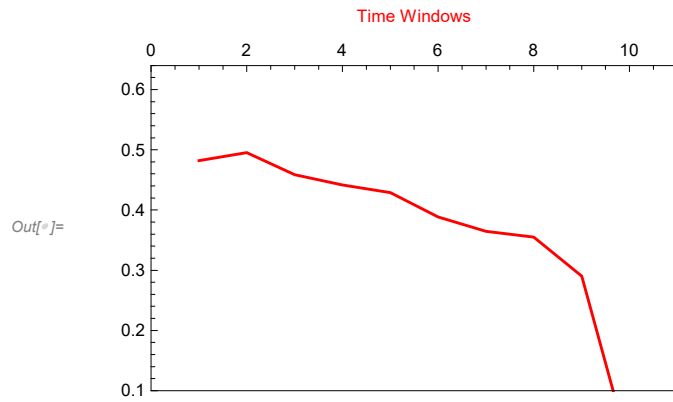
```

```

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]

```

 **ListLinePlot:** Value of option PlotRange → {{0, 11}, {Indeterminate, Indeterminate}} is not All, Full, Automatic, a positive machine number, or an appropriate list of range specifications.



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

```
In[*]:= AbsoluteTiming[widthdataFixedbucket2 =  

  Table[snetworkdatafxdbucket[3, bucketnode12[[i]], datafulllist[[i]]], {i, 10}];]
```

Out[*]= {3.0991, 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]]}];]

```

... Power: Infinite expression $\frac{1}{0}$ encountered.

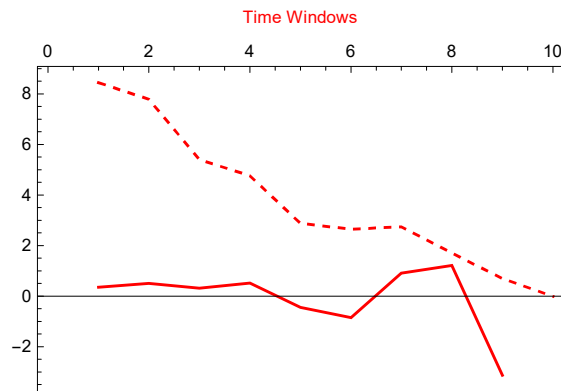
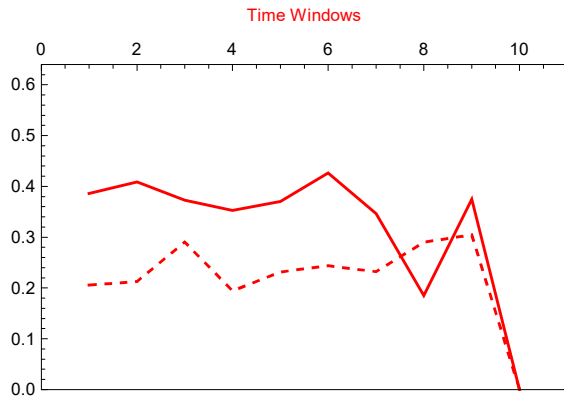
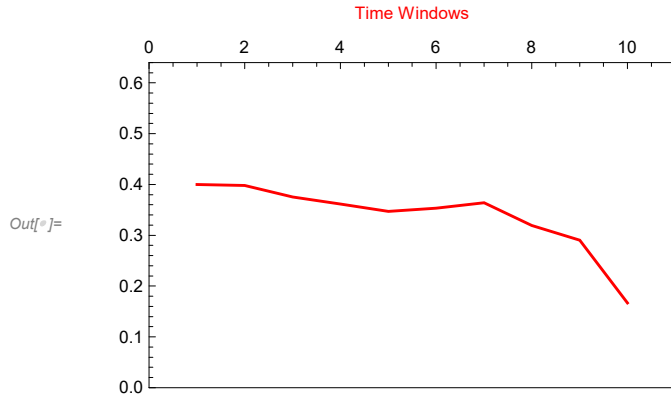
```
Out[ ]:= {55.6433, Null}
```

```
In[ ]:= modularityvaluestimewinsmall = modularityvalues32;
randommodtimewinsmalldegreefxd = 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, 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]
```

... Min: Invalid comparison with ComplexInfinity attempted.

... Max: Invalid comparison with ComplexInfinity attempted.

... ListLinePlot: Value of option PlotRange → {{0, 11}, {-1 + Min[-3.13343, ComplexInfinity], 1 + Max[8.44128, ComplexInfinity]}} is not All, Full, Automatic, a positive machine number, or an appropriate list of range specifications.



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

```

In[ ]:= Export["plot_values/fxd_" <> case <> "/" <> interval <>
  "_" <> interval2 <> "-modularityvalues-fss.mx", modularityvalues12]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <> interval2 <>
  "-singrand-erd-modularityvalues-fss.mx", singlerandomerdrenmodularityvalues12]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <> interval2 <>
  "-singrand-comm-modularityvalues-fss.mx", singlerandomcommmodularityvalues12]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <> interval2 <> "-zscores-fss.mx",
  Zscoresmodularity12]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <>
  interval2 <> "-modularityvalues-fbs.mx", modularityvalues32]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <> interval2 <>
  "-singrand-erd-modularityvalues-fbs.mx", singlerandomerdrenmodularityvalues32]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <> interval2 <>
  "-singrand-comm-modularityvalues-fbs.mx", singlerandomcommmodularityvalues32]
Export["plot_values/fxd_" <> case <> "/" <> interval <> "_" <> interval2 <> "-zscores-fbs.mx",
  Zscoresmodularity32]

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-modularityvalues-fss.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-singrand-erd-
  modularityvalues-fss.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-singrand-comm-
  modularityvalues-fss.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-zscores-fss.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-modularityvalues-fbs.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-singrand-erd-
  modularityvalues-fbs.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-singrand-comm-
  modularityvalues-fbs.mx

Out[ ]:= plot_values/fxd_bounds/25percentdecreased_(-1,1)_-250+250_105-zscores-fbs.mx

```