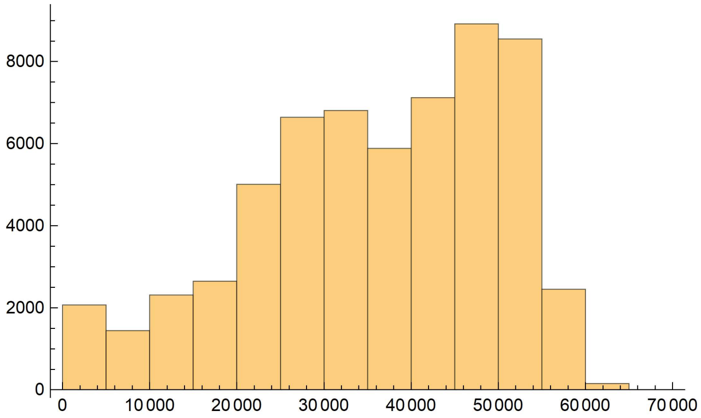
```
In[∘]:= SetDirectory[
          "C:/Users/serha/OneDrive/Masaüstü/MyRepo/master_thesis_MMT003/210507_time_windows_and
            _OR_model"];

In[∘]:= Get["../algoritm_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[∘]:= (* coefficients=
          Table[Table[RandomReal[{-1,1},Length@i],300],{i,subsetpositionsforsequences}]; *)

In[∘]:= (* boundariesposdouble=
         Join[RandomSample[Complement[Range@fluxexchanges,boundariespos],105],boundariespos];
        Export["boundariesposdouble.mx",boundariesposdouble] *)

In[∘]:= subsetpositionsforsequences = Import["subsetpositionsforsequences.mx"];
        boundariesposdouble = Import["boundariesposdouble.mx"];
        coefficients = Import["coefficients.mx"];

In[∘]:= boundariesd = ReplacePart[ConstantArray[{-500, 500}, fluxexchanges], MapThread[#1 → #2 &,
            {boundariesposdouble, ConstantArray[{-50, 50}, Length@boundariesposdouble]}]];

In[∘]:= syntheticseqgenerator[stoichiometricmatrix_, steadystatevector_,
          boundaries_, fluxexchanges_, subsetpositions_, coefficients_] :=
         Module[{objectivefunctions, solutionvectors},
          objectivefunctions = Table[ReplacePart[ConstantArray[0., fluxexchanges],
             MapThread[#1 → #2 &, {subsetpositions, coefficients[[i]]}]], {i, 300}];
          solutionvectors = Chop[Table[LinearProgramming[-objectivefunctions[[i]],
              stoichiometricmatrix, steadystatevector, boundaries],
             {i, Length@objectivefunctions}], 10^-5];
          {objectivefunctions, solutionvectors, MapThread[Dot,
            {objectivefunctions, solutionvectors}]}]
```

```
In[∘]:= AbsoluteTiming[objfuncsforsequences = Table[syntheticseqgenerator[stoichiometricmatrix,
             steadystatevector, boundariesd, fluxexchanges, i[[1]], i[[2]]],
           {i, MapThread[{#1, #2} &, {subsetpositionsforsequences, coefficients}]}];]

Out[∘]= {5722.8, Null}
```

```
In[∘]:= Length@first[(Flatten[objfuncsforsequences[[All, 2]], 1])^T]
        Length@(Flatten[objfuncsforsequences[[All, 2]], 1])[[
          first@Flatten[objfuncsforsequences[[All, 2]], 1], All]]

Out[∘]= 1008

Out[∘]= 60 000
```

```
In[∘]:= datafull = Join[Partition[Range@60 000, 1],
           Partition[Flatten@Table[ConstantArray[i, 300], {i, 200}], 1],
           Partition[Flatten[objfuncsforsequences[[All, 3]], 1], 1], 2];
        Histogram@datafull[[All, 3]]
```

Out[∘]=



```
In[∘]:= x2 = Round@Ceiling[Length@datafull / 19, 1];
        {a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, r, s, t} =
          Join[Range[x2, Length@datafull, x2], {Length@datafull}];
        data2 = Join[{Take[datafull, {1, a}]},
           Flatten[Table[{Take[datafull, {z[[1]] - x2 / 2, z[[2]] - x2 / 2}],
              Take[datafull, {z[[1]], z[[2]]}]}, {z,
              Partition[{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, r, s, t}, 2, 1]}], 1]];
        win2 = Length@data2;
```

```
In[∘]:= AbsoluteTiming[
         widthdataintimewindowsFixedstep2 = snetworkdatabinnedintimewindows[data2, 3, 1000, win2];]

Out[∘]= {30.2504, Null}
```

```
In[∘]:= graphsandnodenumbers12 = Table[snetworkgraph[widthdataintimewindowsFixedstep2[[1]][[i]],
            widthdataintimewindowsFixedstep2[[2]][[i]], 2, 7, 400, Green], {i, Range@win2}];
        graphsandnodenumbers12[[All, 2]]

Out[∘]= {49, 43, 44, 51, 52, 52, 49, 52, 54, 47, 44, 53, 49, 46, 51, 47, 56, 48,
         52, 45, 43, 43, 39, 38, 41, 43, 52, 49, 50, 41, 53, 49, 39, 41, 32, 53, 54}
```

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

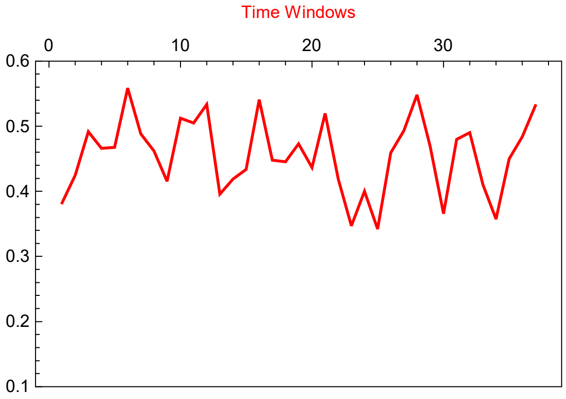
```
In[∘]:= singlerandomgraphsdegfxd12 =
          Table[randomizinggraphdegfxd[i], {i, graphsandnodenumbers12[[All, 1]]}];
        singlerandomerdrenmodularityvalues12 =
          Table[N@GraphAssortativity[singlerandomgraphsdegfxd12[[i]],
             FindGraphCommunities[singlerandomgraphsdegfxd12[[i]]], "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}];
```

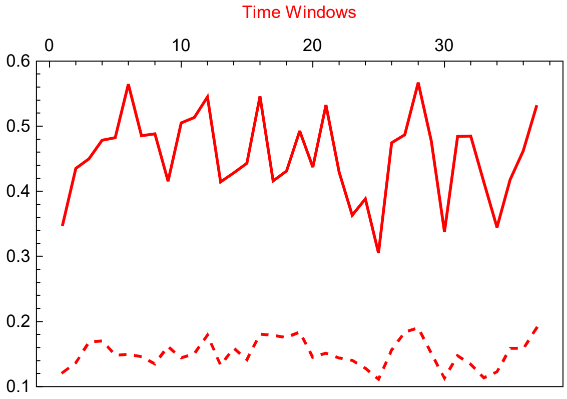
```
In[∘]:= AbsoluteTiming[Zscoresmodularity12 =
           Table[zscorefunctionfortwonullmodels[i], {i, graphsandnodenumbers12[[All, 1]]}];]

Out[∘]= {530.114, Null}
```

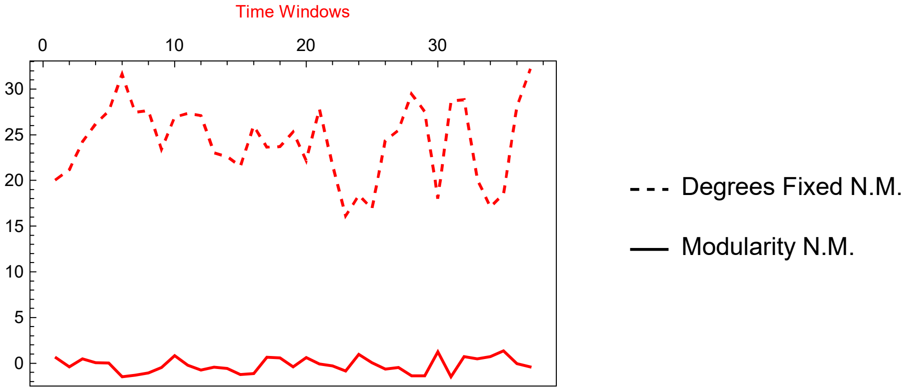
```
In[∘]:= bucketnode12 = graphsandnodenumbers12[[All, 2]]

Out[∘]= {49, 43, 44, 51, 52, 52, 49, 52, 54, 47, 44, 53, 49, 46, 51, 47, 56, 48,
         52, 45, 43, 43, 39, 38, 41, 43, 52, 49, 50, 41, 53, 49, 39, 41, 32, 53, 54}
```

```
In[•]:= modularityvaluestimewinsmall = modularityvalues12;
randommodtimewinsmalldegreefxd = singlerandomerdrenmodularityvalues12;
randommodtimewinsmallcomm = singlerandomcommmodularityvalues12;
Zscoretimewinsmall = Zscoresmodularity12;
modularityplotrange = {0.1, 0.6};
(*MinMax[{modularityvalues1,singlerandomcommmodularityvalues1,
  singlerandomerdrenmodularityvalues1,modularityvalues12}]*)
padding = 38;
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 → {{-1, win2 + 2}, 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 → {{-1, win2 + 2}, 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 → {{-1, win2 + 2}, MinMax[Flatten[Zscoretimewinsmall], 1]}]}],
  LineLegend[{Dashed, Black}, {"Degrees Fixed N.M.", "Modularity N.M."},
   LegendMargins → 0, LegendMarkerSize → {20, 20}], Spacer@0.1}]
```

Out[◦]=







```
In[◦]:= AbsoluteTiming[widthdataintimewindowsFixedbucket2 =
           snetworkdatafxdbucketintimewindows[data2, 3, bucketnode12, win2];]

Out[◦]= {2.70053, Null}
```

```
In[ ]:= bucketsize32 = Flatten@widthdataintimewindowsFixedbucket2[[4]]

Out[ ]= {65, 74, 72, 62, 61, 61, 65, 61, 59, 68, 72, 60, 65, 69, 62, 68, 57, 66,
         61, 71, 74, 74, 81, 84, 78, 74, 61, 65, 64, 78, 60, 65, 81, 78, 99, 60, 59}
```

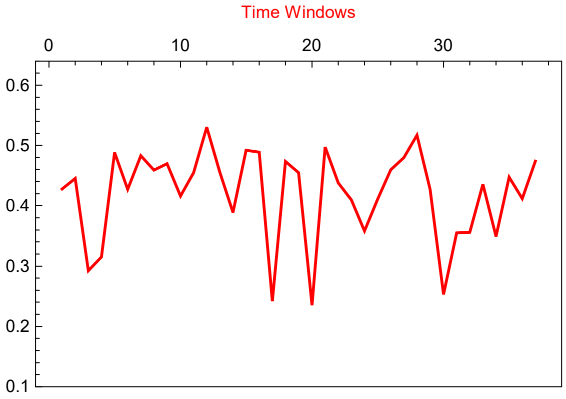
```
In[ ]:= graphsandnodenumbers32 =
          Table[snetworkgraph[widthdataintimewindowsFixedbucket2[[1]][[i]],
            widthdataintimewindowsFixedbucket2[[2]][[i]], 1.5, 7, 400, Green], {i, Range@win2}];
        modularityvalues32 = Table[N@GraphAssortativity[graphsandnodenumbers32[[i]][[1]],
              FindGraphCommunities[graphsandnodenumbers32[[i]][[1]]], "Normalized" → False],
            {i, Length@graphsandnodenumbers32}];
```

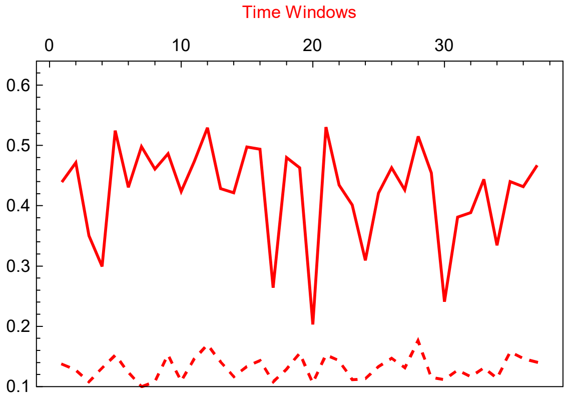
```
In[ ]:= singlerandomgraphsdegfxd32 =
          Table[randomizinggraphdegfxd[i], {i, graphsandnodenumbers32[[All, 1]]}];
        singlerandomerdrenmodularityvalues32 =
          Table[N@GraphAssortativity[singlerandomgraphsdegfxd32[[i]],
              FindGraphCommunities[singlerandomgraphsdegfxd32[[i]]], "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}];
```

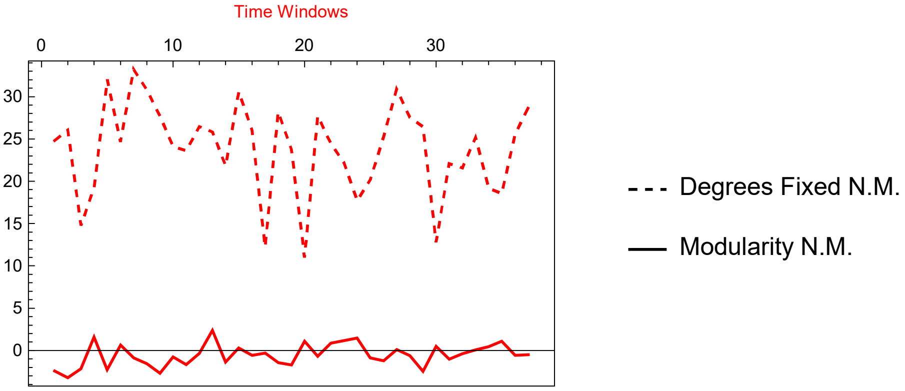
```
In[ ]:= AbsoluteTiming[Zscoresmodularity32 =
           Table[zscorefunctionfortwonullmodels[i], {i, graphsandnodenumbers32[[All, 1]]}];]

Out[ ]= {420.707, Null}
```

```
In[•]:= modularityvaluestimewinsmall = modularityvalues32;
        randommodtimewinsmalldegreefxd = singlerandomerdrenmodularityvalues32;
        randommodtimewinsmallcomm = singlerandomcommmodularityvalues32;
        Zscoretimewinsmall = Zscoresmodularity32;
        modularityplotrange = {0.1, 0.64};
        (*MinMax[{modularityvalues1,singlerandomcommmodularityvalues1,
          singlerandomerdrenmodularityvalues1,modularityvalues12}]*)
        padding = 38;
        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 → {{-1, win2 + 2}, 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 → {{-1, win2 + 2}, 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 → {{-1, win2 + 2}, MinMax[Flatten[Zscoretimewinsmall], 1]}]}],
          LineLegend[{Dashed, Black}, {"Degrees Fixed N.M.", "Modularity N.M."},
            LegendMargins → 0, LegendMarkerSize → {20, 20}], Spacer@0.1}]
```

Out[∘]=


Time Windows
0
10
20
30
0.6
0.5
0.4
0.3
0.2
0.1


Time Windows
0
10
20
30
0.6
0.5
0.4
0.3
0.2
0.1


Time Windows
0
10
20
30
30
25
20
15
10
5
0
- - - Degrees Fixed N.M.
— Modularity N.M.

```
In[ ]:= Export["plot_values/boundariesdouble_(-50,50)-modularityvalues-fss.mx",
         modularityvalues12]
        Export["plot_values/boundariesdouble_(-50,50)-singrand-erd-modularityvalues-fss.mx",
         singlerandomerdrenmodularityvalues12]
        Export["plot_values/boundariesdouble_(-50,50)-singrand-comm-modularityvalues-fss.mx",
         singlerandomcommmodularityvalues12]
        Export["plot_values/boundariesdouble_(-50,50)-zscores-fss.mx", Zscoresmodularity12]
        Export["plot_values/boundariesdouble_(-50,50)-modularityvalues-fbs.mx",
         modularityvalues32]
        Export["plot_values/boundariesdouble_(-50,50)-singrand-erd-modularityvalues-fbs.mx",
         singlerandomerdrenmodularityvalues32]
        Export["plot_values/boundariesdouble_(-50,50)-singrand-comm-modularityvalues-fbs.mx",
         singlerandomcommmodularityvalues32]
        Export["plot_values/boundariesdouble_(-50,50)-zscores-fbs.mx", Zscoresmodularity32]

Out[ ]= plot_values/boundariesdouble_(-50,50)-modularityvalues-fss.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-singrand-erd-modularityvalues-fss.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-singrand-comm-modularityvalues-fss.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-zscores-fss.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-modularityvalues-fbs.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-singrand-erd-modularityvalues-fbs.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-singrand-comm-modularityvalues-fbs.mx

Out[ ]= plot_values/boundariesdouble_(-50,50)-zscores-fbs.mx
```