

Мухамадиев Владимир

Задание 4

Загрузка и предварительная обработка

```
In[1]:= data1 = Import[NotebookDirectory[] <> "\\bio-celegans.txt", "Data"];
          |импорт |директория файла блокнота

In[2]:= el1 = Table[data1[[i, 1] → data1[[i, 2]], {i, 1, Length[data1]}];
          |таблица значений |длина

In[3]:= g1 = Graph[el1];
          |граф

In[4]:= data2 = Import[NotebookDirectory[] <> "\\bio-diseasome.txt", "Data"];
          |импорт |директория файла блокнота

In[5]:= el2 = Table[data2[[i, 1] → data2[[i, 2]], {i, 1, Length[data2]}];
          |таблица значений |длина

In[6]:= g2 = Graph[el2];
          |граф

In[7]:= data3 = Import[NotebookDirectory[] <> "\\cit-HepTh.txt", "Data"];
          |импорт |директория файла блокнота

In[8]:= el3 = Table[data3[[i, 1] → data3[[i, 2]], {i, 1, Length[data3]}];
          |таблица значений |длина

In[9]:= g3 = Graph[el3];
          |граф

In[10]:= MinMaxNormalization[data_] := Table[(data[[i]] - Min[data]) / (Max[data] - Min[data]), {i, 1, Length[data]}]
          |таблица значений |длина
```

1. Complementary cumulative distribution function (Survival function)

```
In[11]:= CCDF[arr_, nbins_] := Table[N[{i, SurvivalFunction[EmpiricalDistribution[arr], i]}], {i, Min[arr], Max[arr], (Max[arr] - Min[arr]) / nbins - 1}]
          |таблица |числ... |функция выживания |эмпирическое распределение |минимум |максимум |длина
```

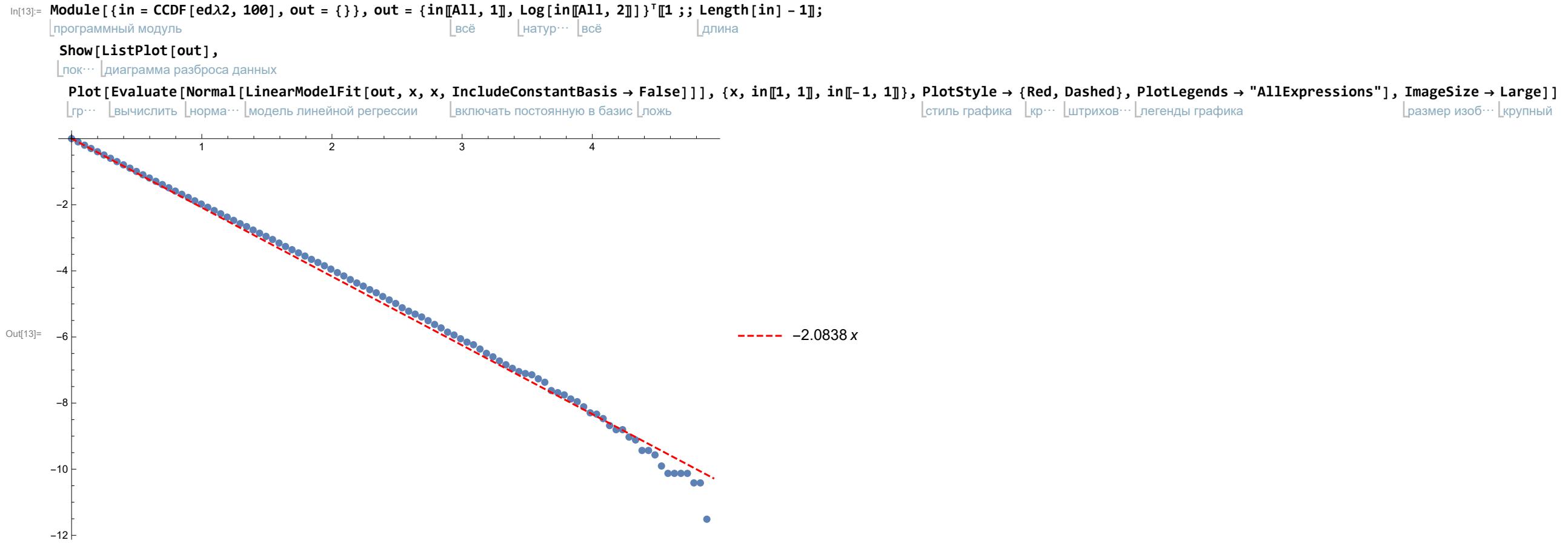
Экспоненциальное распределение

$$\begin{aligned} \text{PDF}(x) &= \lambda e^{-\lambda x} \\ \text{CDF}(x) &= 1 - e^{-\lambda x} \\ \text{CCDF}(x) &= e^{-\lambda x} \\ \ln(\text{CCDF}(x)) &= -\lambda x \end{aligned}$$

При отображении $x \rightarrow \ln(\text{CCDF}(x))$, получаем линейную зависимость вида $f(x) = -ax$, где $a = \lambda$

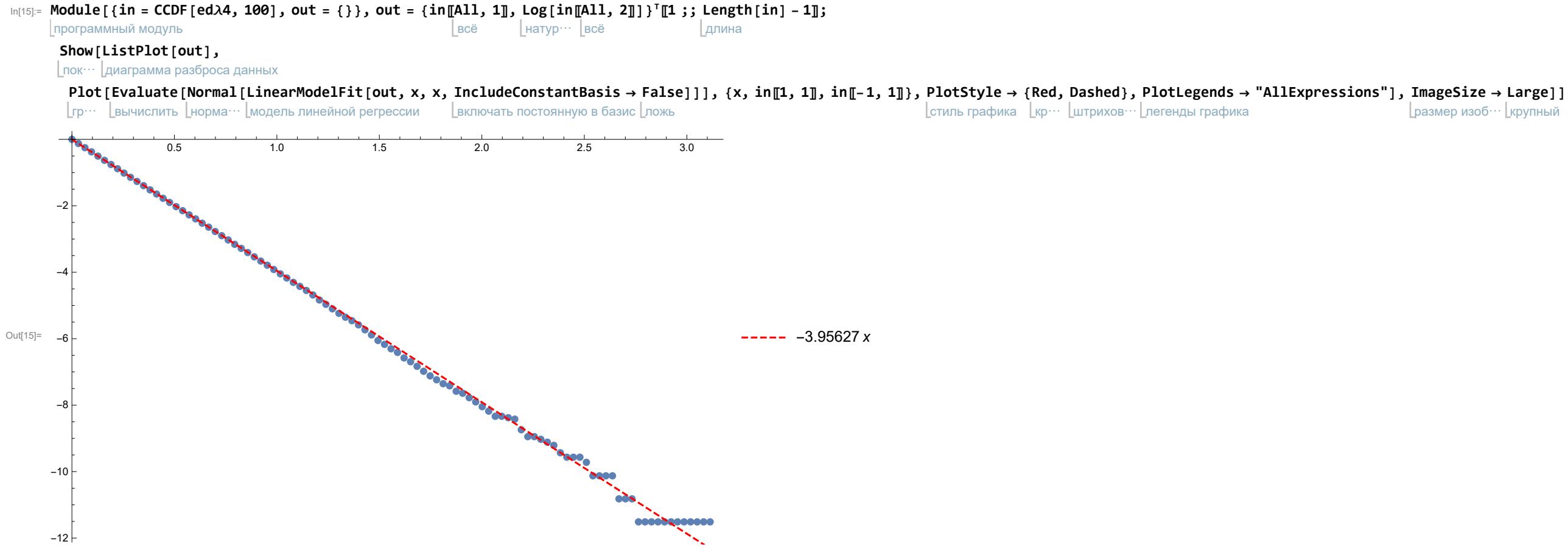
$$\lambda=2$$

```
In[12]:= edλ2 = RandomVariate[ExponentialDistribution[2], 100000];
          |реализация слу... |показательное распределение
```

 $\lambda=4$

In[14]:= `edλ4 = RandomVariate[ExponentialDistribution[4], 100000];`

реализация слу... показательное распределение



Степенное распределение

$$\text{PDF}(x) = \frac{k(x_{\min})^k}{x^{k+1}}$$

$$\text{CDF}(x) = 1 - \left(\frac{x_{\min}}{x}\right)^k$$

$$\text{CCDF}(x) = \left(\frac{x_{\min}}{x}\right)^k$$

$$\ln(\text{CCDF}(x)) = k \ln(x_{\min}) - k \ln(x)$$

При отображении $\ln(x) \rightarrow \ln(\text{CCDF}(x))$, получаем линейную зависимость вида $f(x) = -ax + b$, где $a = k$

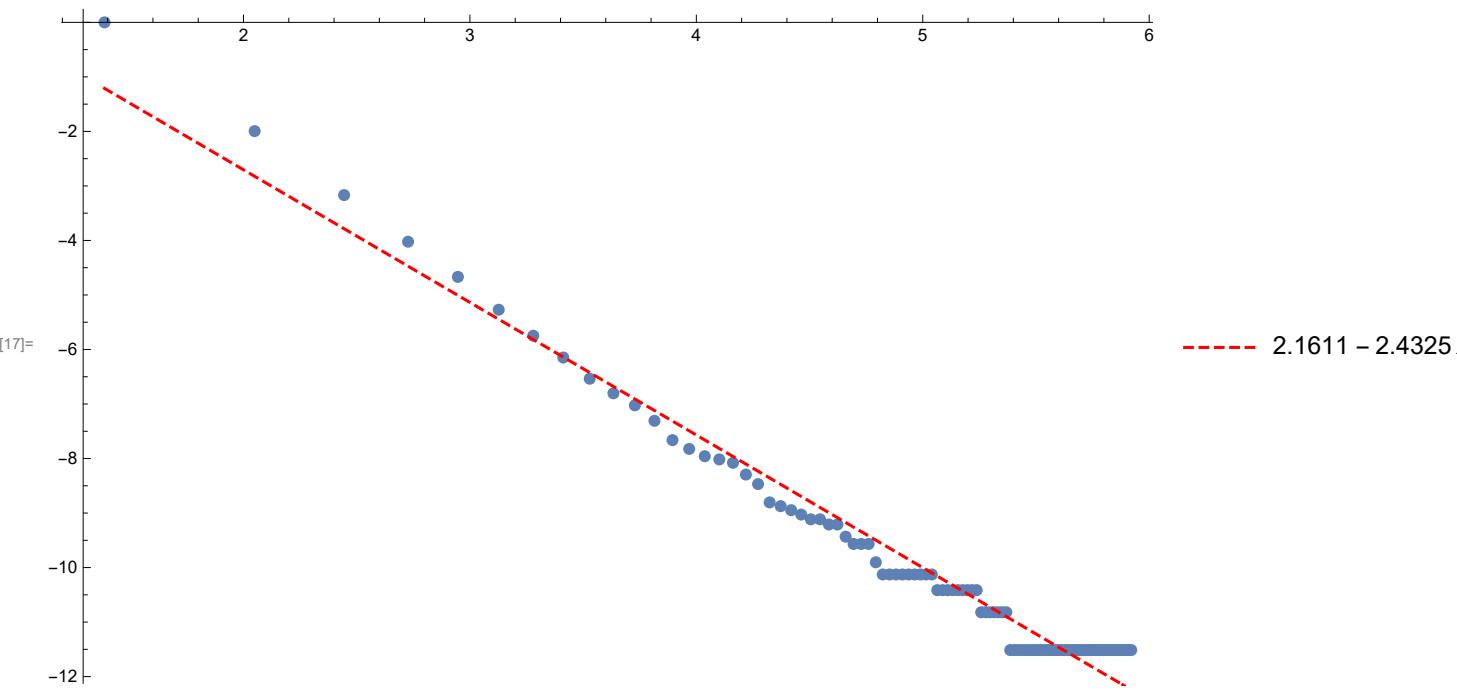
$$x_{\min} = 4, k = 3$$

```
In[16]:= pdxm4k3 = RandomVariate[ParetoDistribution[4, 3], 100000];

```

```
In[17]:= Module[{in = CCDF[pdxm4k3, 100], out = {}}, out = Log[in][[1;; Length[in] - 1]];
Show[ListPlot[out], Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], ImageSize -> Large]]
Out[17]=
```

пок... диаграмма разб... [гра... вычислить [норма... модель линейной регрессии] стиль графика [кру... штрихов... легенды графика] размер изоб... крупный



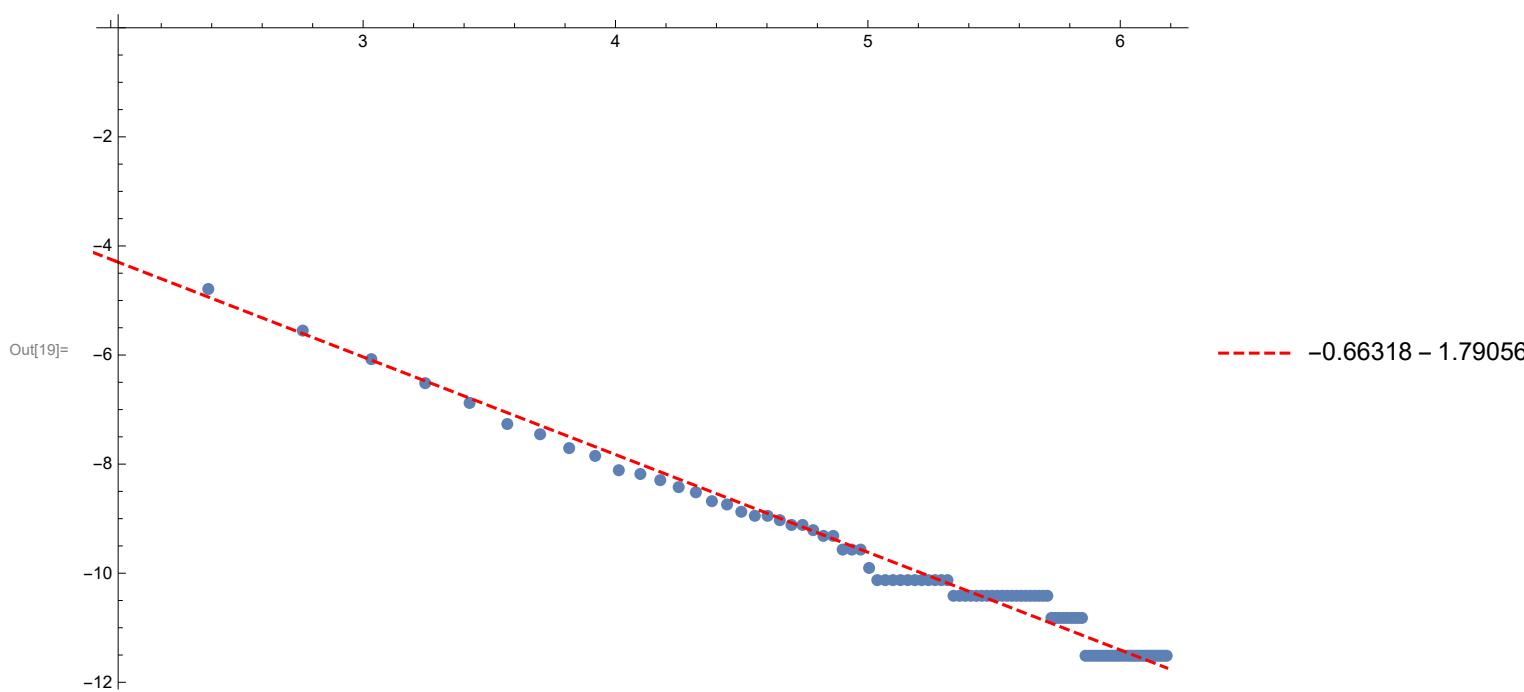
$$x_{\min} = 1, k = 2$$

```
In[18]:= pdxm1k2 = RandomVariate[ParetoDistribution[1, 2], 100000];
Show[ListPlot[out], Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], ImageSize -> Large]]
Out[18]=
```

реализация слу... распределение Парето

```
In[19]:= Module[{in = CCDF[pdxm1k2, 100], out = {}}, out = Log[in][[1;; Length[in] - 1]];
Show[ListPlot[out], Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], ImageSize -> Large]]
Out[19]=
```

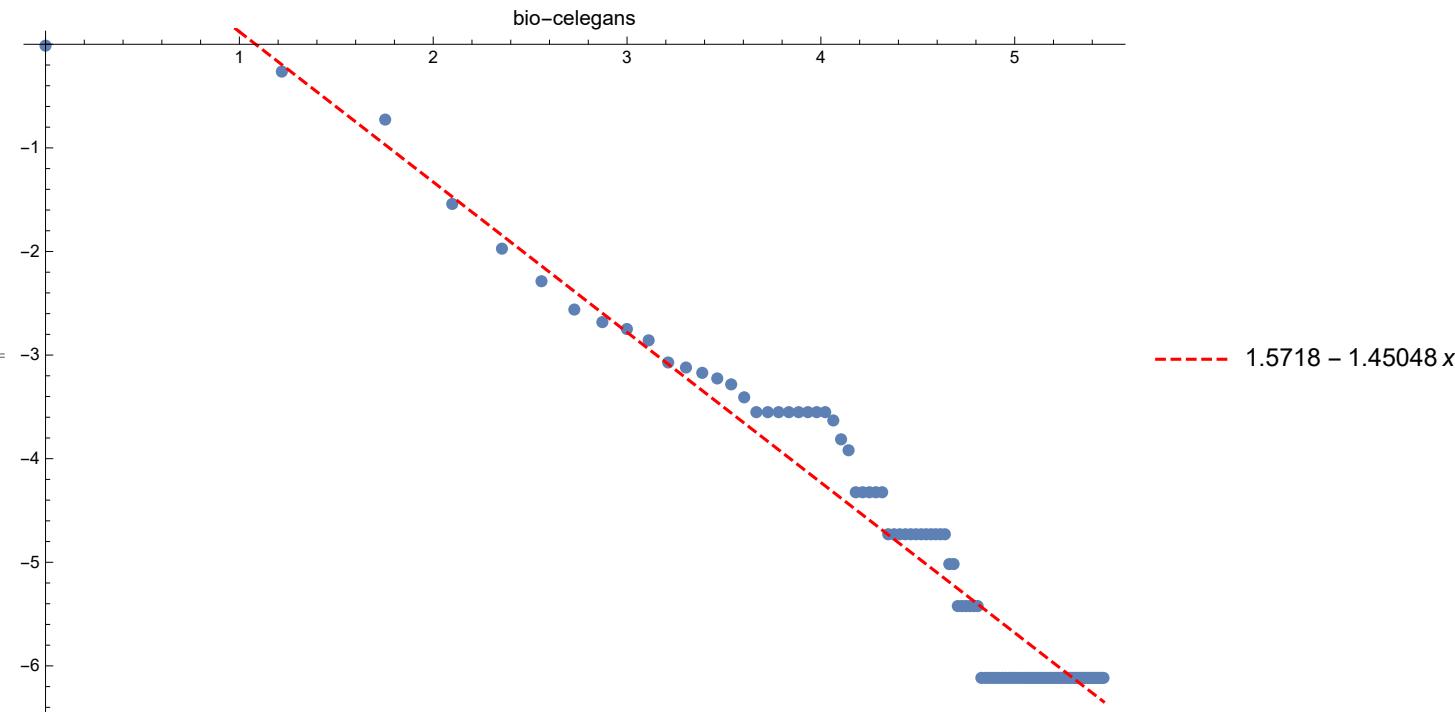
пок... диаграмма разб... [гра... вычислить [норма... модель линейной регрессии] стиль графика [кру... штрихов... легенды графика] размер изоб... крупный



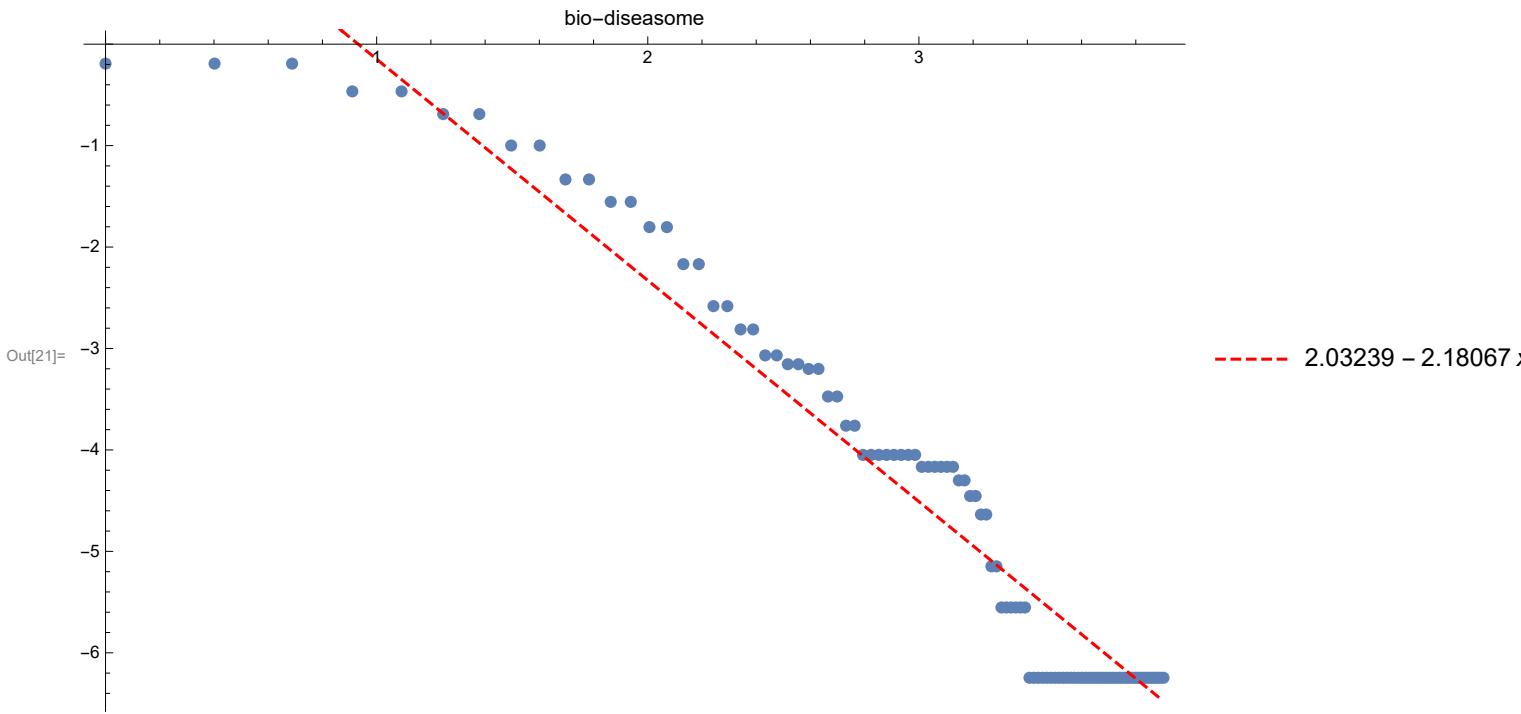
Распределение степеней вершин в заданных сетях

```
In[20]:= Module[{in = CCDF[VertexDegree[g1], 100], out = {}}, out = Log[in][[1;; Length[in] - 1]];
Show[ListPlot[out],
Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions", PlotLabel -> "bio-celegans", ImageSize -> Large]]
Out[20]=
```

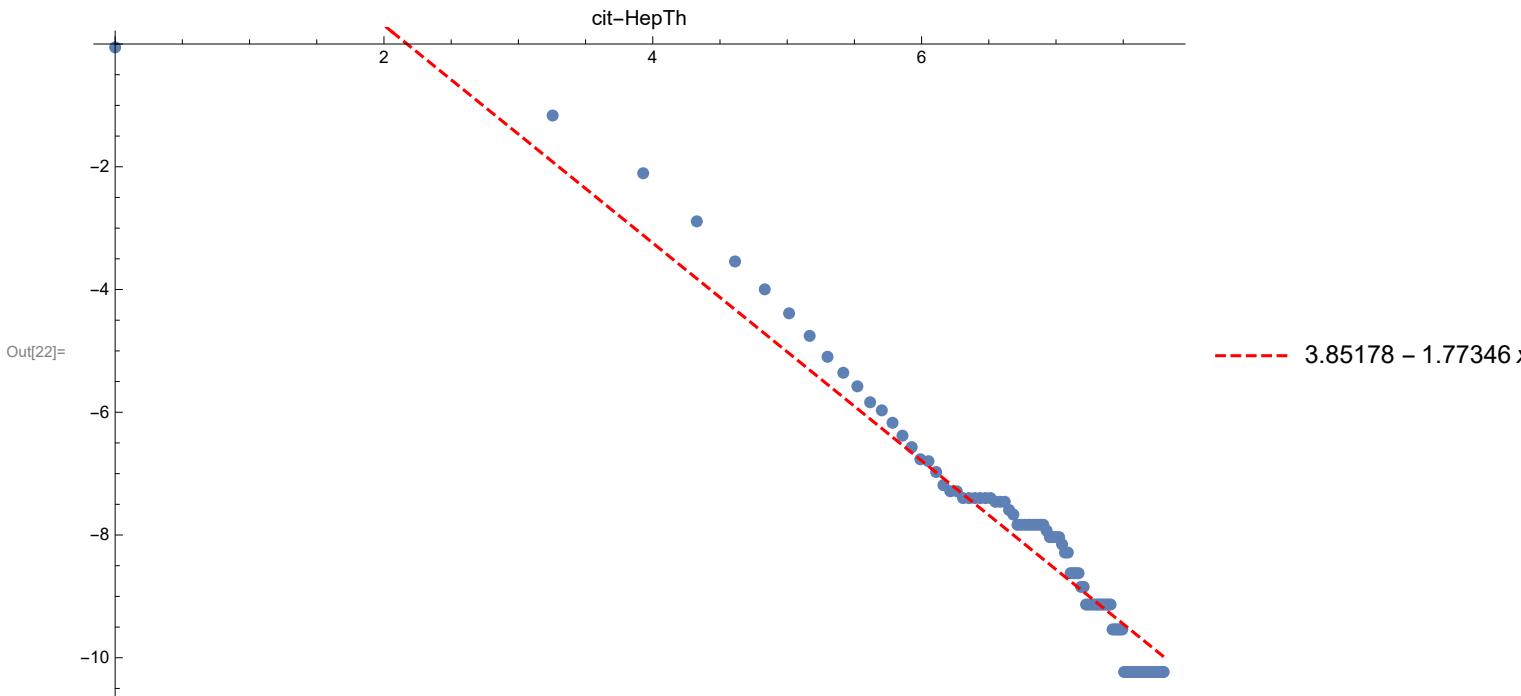
Module[$\{in = CCDF[VertexDegree[g1], 100], out = \{\}\}, out = Log[in][[1;; Length[in] - 1]];$
Show[$ListPlot[out]$,
Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions", PlotLabel -> "bio-celegans", ImageSize -> Large]]



```
In[21]:= Module[{in = CCDF[VertexDegree[g2], 100], out = {}}, out = Log[in][[1 ;; Length[in] - 1]];
  Show[ListPlot[out],
    Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], PlotLabel -> "bio-diseasome", ImageSize -> Large]]
  ]
Out[21]=
```



```
In[22]:= Module[{in = CCDF[VertexDegree[g3], 100], out = {}}, out = Log[in][[1 ;; Length[in] - 1]];
  Show[ListPlot[out],
    Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], PlotLabel -> "cit-HepTh", ImageSize -> Large]]
  ]
Out[22]=
```



Распределение степеней вершин в модели Барабаши-Альберт

$$\text{PDF}(k) = 2m^2 k^{-3}$$

$$\text{CDF}(k) = 1 - m^2 k^{-2}$$

$$\text{CCDF}(k) = m^2 k^{-2}$$

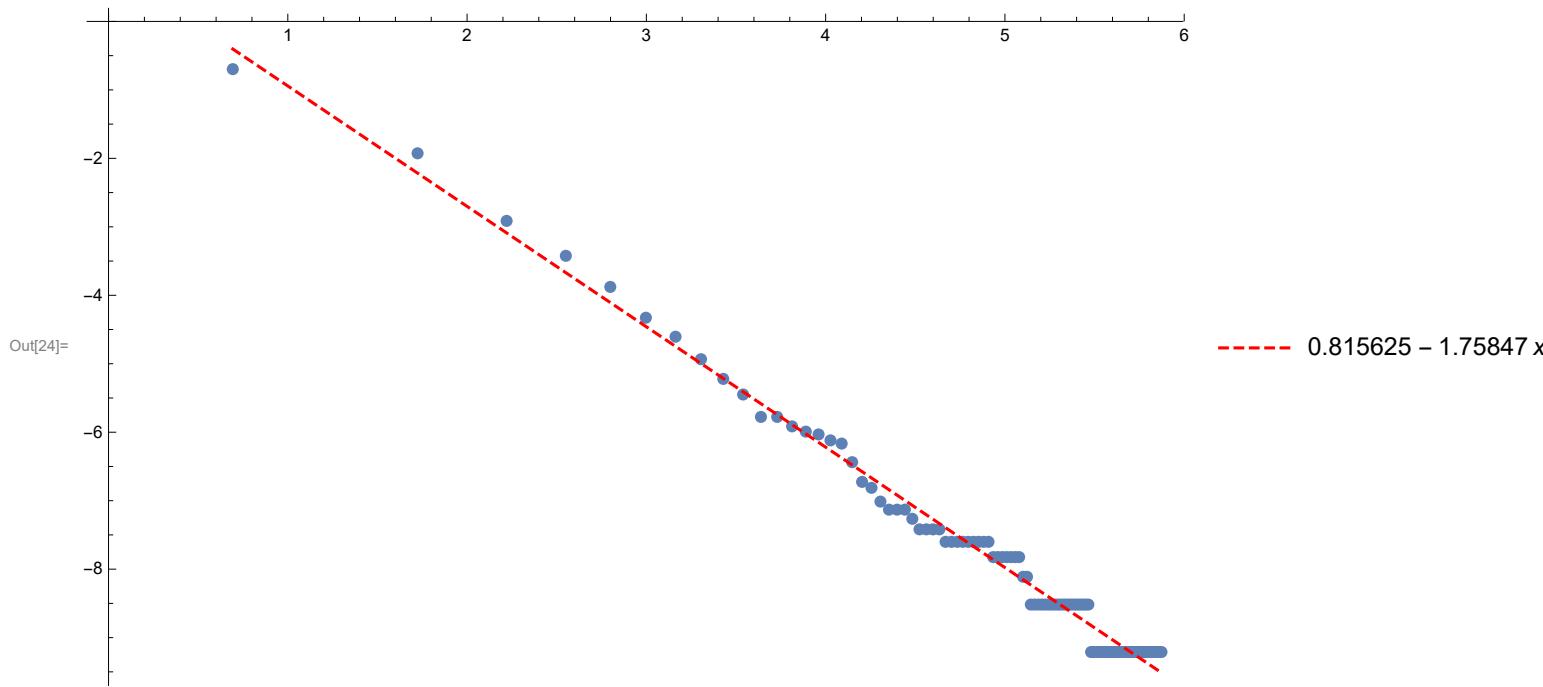
$$\ln(\text{CCDF}(k)) = 2 \ln(m) - 2 \ln(k)$$

При отображении $\ln(k) \rightarrow \ln(\text{CCDF}(k))$, получаем линейную зависимость вида $f(x) = -2x + a$

$m=2$

```
In[23]:= bam2 = RandomGraph[BarabasiAlbertGraphDistribution[10000, 2]];
          [случайный г... [графово распределение Барабаши-Альберт

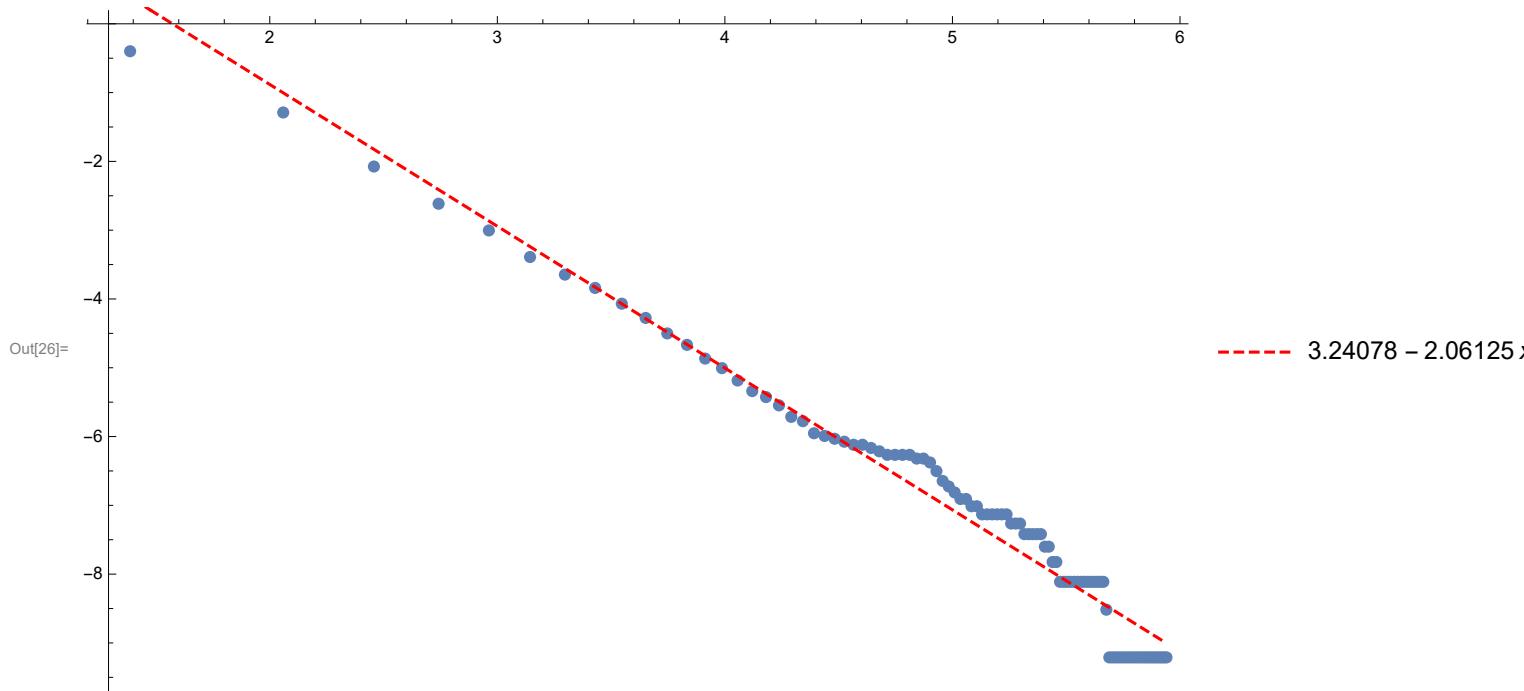
In[24]:= Module[{in = CCDF[VertexDegree[bam2], 100], out = {}}, out = Log[in][[1;; Length[in] - 1]];
          [программный модуль [степень вершины
          Show[ListPlot[out], Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[-1, 1]}], PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], ImageSize -> Large]]
          [пок... [диаграмма разб... [гр... [вычислить [норма... [модель линейной регрессии [стиль графика [кр... [штрихов... [легенды графика [размер изоб... [крупный
```



$m=4$

```
In[25]:= bam4 = RandomGraph[BarabasiAlbertGraphDistribution[10000, 4]];
          [случайный г... [графово распределение Барабаши-Альберт
```

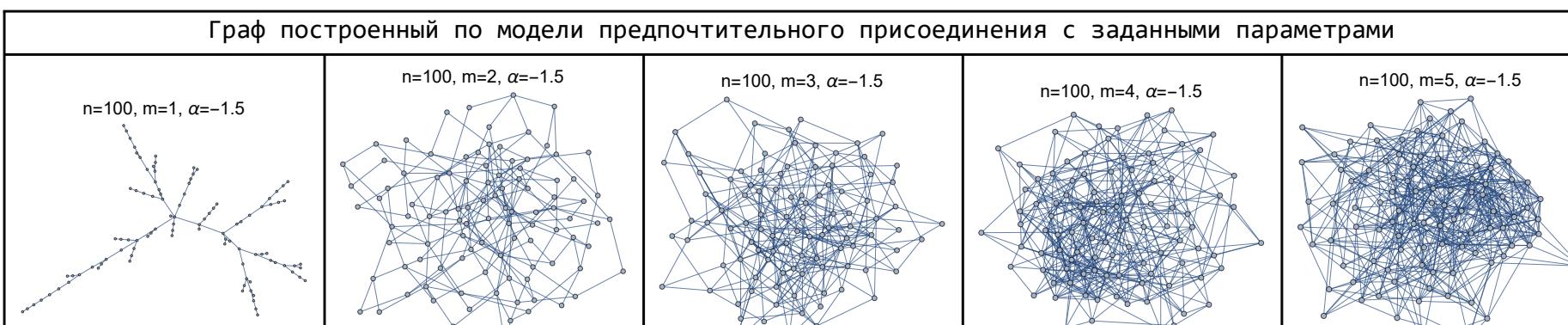
```
In[26]:= Module[{in = CCDF[VertexDegree[bam4], 100], out = {}}, out = Log[in][[1 ;; Length[in] - 1]];
Show[ListPlot[out], Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], ImageSize -> Large]]
Out[26]=
```

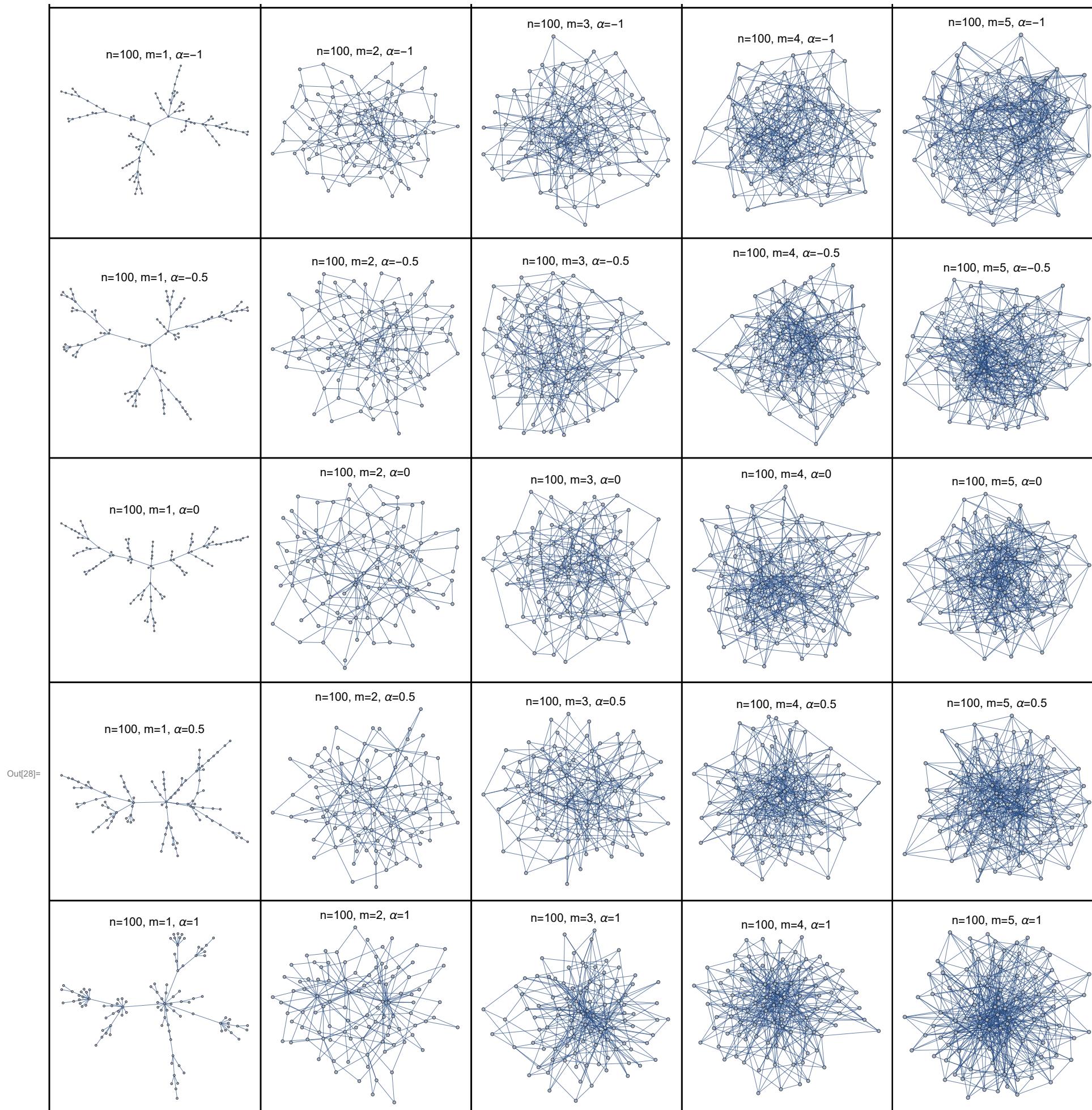


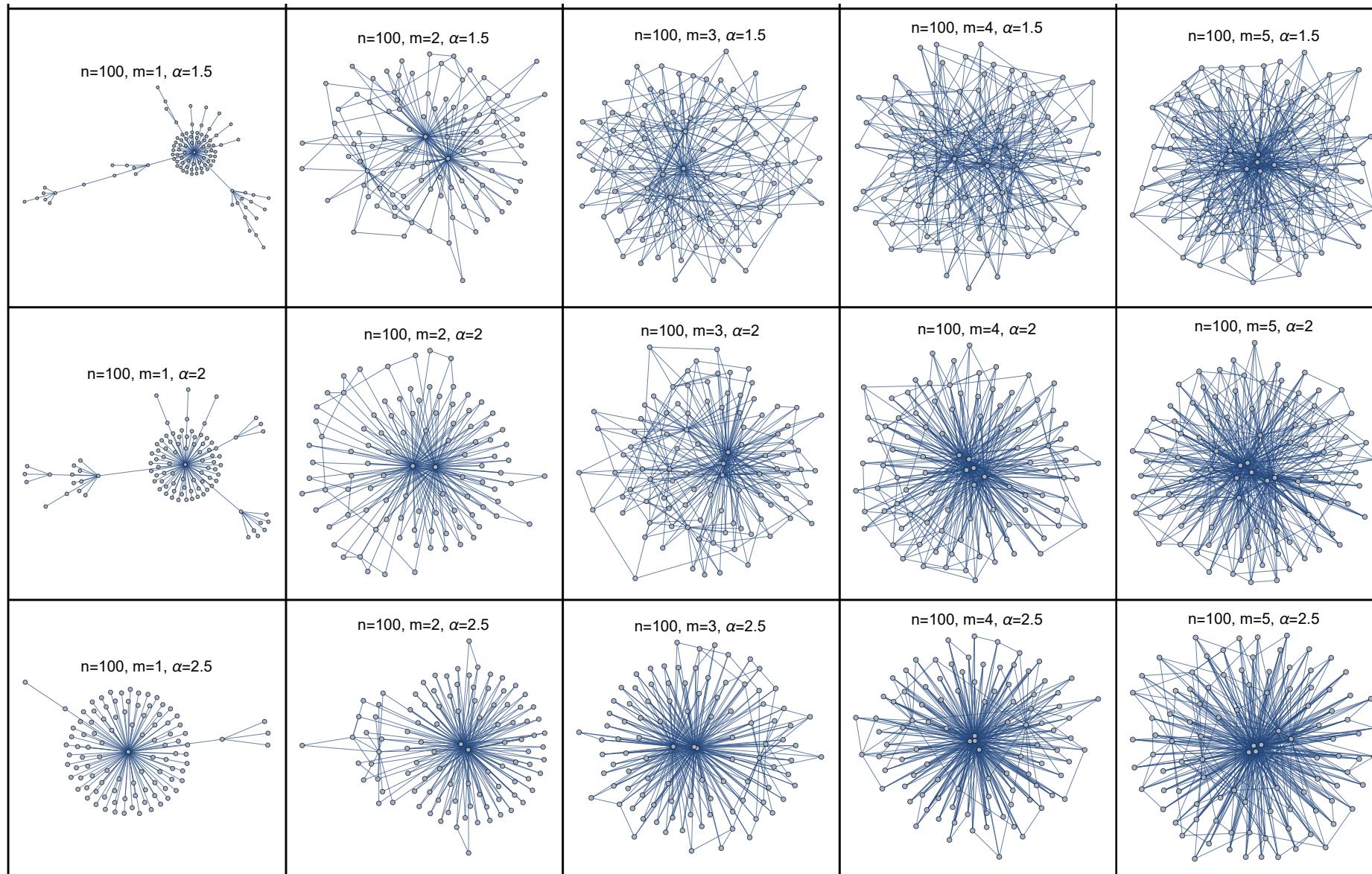
2. Предпочтительное присоединение

Генерация графа по предпочтительному присоединению с заданными параметрами в виде списка ребер

```
In[27]:= BarabasiAlbertGraphEdgeList[n_, m_, α_] := If[n ≤ m, EdgeList[CompleteGraph[n]],  
(*m==1*) If[m == 1, Module[{el = PadRight[{1 → 2}, n - 1, {}]}, pl = {}, vl = Table[i, {i, 1, n}], vla = {}],  
(*m==1 and α==0*) If[α == 0, Do[vla = RandomChoice[vl[[1 ;; i - 1]]]; el[[i - 1]] = vla → i, {i, 3, n}]; el,  
(*m==1 and α==1*) If[α == 1, pl = Table[1, {i, 1, n}]; Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]]; el[[i - 1]] = vla → i; pl[[vla]] = pl[[vla]] + 1, {i, 3, n}]; el,  
(*m>1 and α≠0 and α≠1*) pl = Table[1, {i, 1, n}];  
Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]];  
el[[i - 1]] = vla → i;  
pl[[vla]] = (pl[[vla]]1/α + 1), {i, 3, n}]; el]],  
(*m>1*) Module[{el = PadRight[EdgeList[CompleteGraph[m]], m n -  $\frac{1}{2} m(m+1)$ , {}], k =  $\frac{m(m-1)}{2}$ , pl = {}, vl = Table[i, {i, 1, n}], vla = {}},  
(*m>1 and α==0*) If[α == 0, Do[vla = RandomSample[vl[[1 ;; i - 1]], m]; Do[k++; el[[k]] = vla[[j]] → i, {j, 1, m}], {i, m + 1, n}]; el,  
(*m>1 and α==1*) If[α == 1, pl = PadRight[Table[m - 1, {i, 1, m}], n, m];  
Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];  
Do[k++;  
el[[k]] = vla[[j]] → i;  
pl[[vla[[j]]]] = pl[[vla[[j]]]] + 1, {j, 1, m}], {i, m + 1, n}]; el,  
(*m>1 and α≠0 and α≠1*) pl = PadRight[Table[(m - 1)α, {i, 1, m}], n, mα];  
Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];  
Do[k++;  
el[[k]] = vla[[j]] → i;  
pl[[vla[[j]]]] = (pl[[vla[[j]]]]1/α + 1), {j, 1, m}], {i, m + 1, n}];  
el]]]]]  
  
In[28]:= Grid[Prepend[Table[Graph[BarabasiAlbertGraphEdgeList[100, p1, p2], PlotLabel → "n=100, m=" <> ToString[p1] <> ", α=" <> ToString[If[p2 ∈ Integers, p2, N[p2]]], ImageSize → Small], {p2, - $\frac{3}{2}$ ,  $\frac{5}{2}$ ,  $\frac{1}{2}$ },  
{p1, 1, 5, 1}], {Style["Граф построенный по модели предпочтительного присоединения с заданными параметрами", 16], SpanFromLeft}], Dividers → All, Spacings → {1, 1}]
```



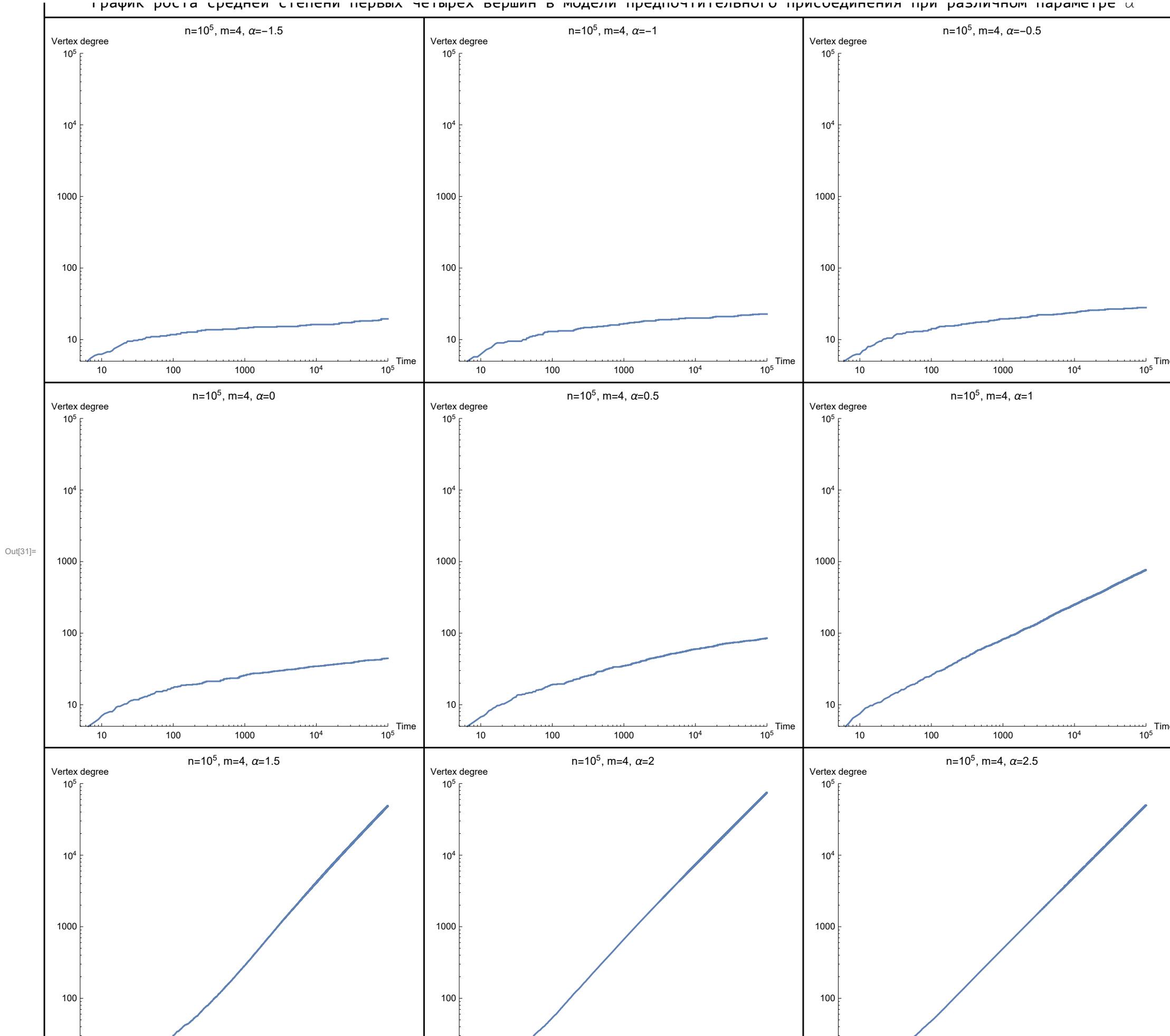


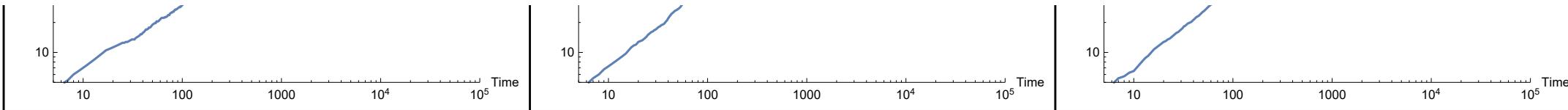


Генерация эволюции роста степени первых m вершин в модели предпочтительного присоединения с заданными параметрами

```
In[29]:= BarabasiAlbertGraphVertexDegreeEvolution[n_, m_, α_] := If[n ≤ m, PadRight[Table[VertexDegree[CompleteGraph[i]], {i, 1, n}]]^T,
(*m==1*) If[m == 1, Module[{vd = PadRight[{0, 1}, n, {}]}, pl = {}, vl = Table[i, {i, 1, n}], vla = {}},
(*m==1 and α==0*) If[α == 0, Do[vla = RandomChoice[vl[[1 ;; i - 1]]]; If[vla == 1, vd[[i]] = vd[[i - 1]] + 1, vd[[i]] = vd[[i - 1]], {i, 3, n}]; vd,
(*m==1 and α==1*) If[α == 1, pl = Table[1, {i, 1, n}];
Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]];
If[vla == 1, vd[[i]] = vd[[i - 1]] + 1, vd[[i]] = vd[[i - 1]]];
pl[[vla]] = pl[[vla]] + 1, {i, 3, n}]; vd,
(*m>1 and α≠0 and α≠1*) pl = Table[1, {i, 1, n}];
Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]];
If[vla == 1, vd[[i]] = vd[[i - 1]] + 1, vd[[i]] = vd[[i - 1]]];
pl[[vla]] = (pl[[vla]]1/α + 1), {i, 3, n}]; vd]];
(*m>1*) Module[{vd = PadRight[PadRight[Table[VertexDegree[CompleteGraph[i]], {i, 1, m}]], n, {ConstantArray[{}, m]}]^T, pl = {}, vl = Table[i, {i, 1, n}], vla = {}},
(*m>1 and α==0*) If[α == 0, pl = PadRight[Table[m - 1, {i, 1, m}], n, m];
Do[vla = Sort[RandomSample[vl[[1 ;; i - 1]], m]];
Do[pl[[vla[[j]]]] = pl[[vla[[j]]]] + 1;
vd[[j, i]] = pl[[j]], {j, 1, m}], {i, m + 1, n}]; vd,
(*m>1 and α==1*) If[α == 1, pl = PadRight[Table[m - 1, {i, 1, m}], n, m];
Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];
Do[pl[[vla[[j]]]] = pl[[vla[[j]]]] + 1;
vd[[j, i]] = pl[[j]], {j, 1, m}], {i, m + 1, n}]; vd,
(*m>1 and α≠0 and α≠1*) pl = PadRight[Table[(m - 1)α, {i, 1, m}], n, mα"]];
Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];
Do[pl[[vla[[j]]]] = (pl[[vla[[j]]]]1/α + 1);
vd[[j, i]] = pl[[j]]1/α, {j, 1, m}], {i, m + 1, n}]; vd]];
(*m>1*) Grid[Prepend[Partition[Table[ListLogLogPlot[{Range[105], Mean[simdata[[i]]]}^T, Joined → True, PlotLabel → "n=105, m=4, α=" <> {"-1.5", "-1", "-0.5", "0", "0.5", "1", "1.5", "2", "2.5"}[[i - 4]], AxesLabel → {"Time", "Vertex degree"}, ImageSize → Medium, PlotRange → {{5, 105}, {5, 105}}, AspectRatio → 1], {i, 5, 13}], 3], {Style["График роста средней степени первых четырех вершин в модели предпочтительного присоединения при различном параметре α", 16], Dividers → All, Spacings → {1, 1}]}
In[30]:= simdata = Import[NotebookDirectory[] <> "\\sim1.m"]; (*BarabasiAlbertGraphVertexDegreeEvolution[100000,4,α],α∈{-3,3,0.5}*)
In[31]:= Grid[Prepend[Partition[Table[ListLogLogPlot[{Range[105], Mean[simdata[[i]]]}^T, Joined → True, PlotLabel → "n=105, m=4, α=" <> {"-1.5", "-1", "-0.5", "0", "0.5", "1", "1.5", "2", "2.5"}[[i - 4]], AxesLabel → {"Time", "Vertex degree"}, ImageSize → Medium, PlotRange → {{5, 105}, {5, 105}}, AspectRatio → 1], {i, 5, 13}], 3], {Style["График роста средней степени первых четырех вершин в модели предпочтительного присоединения при различном параметре α", 16], Dividers → All, Spacings → {1, 1}]}]
```

График роста средней степени вершины в модели предложенной в приложении при различных параметрах



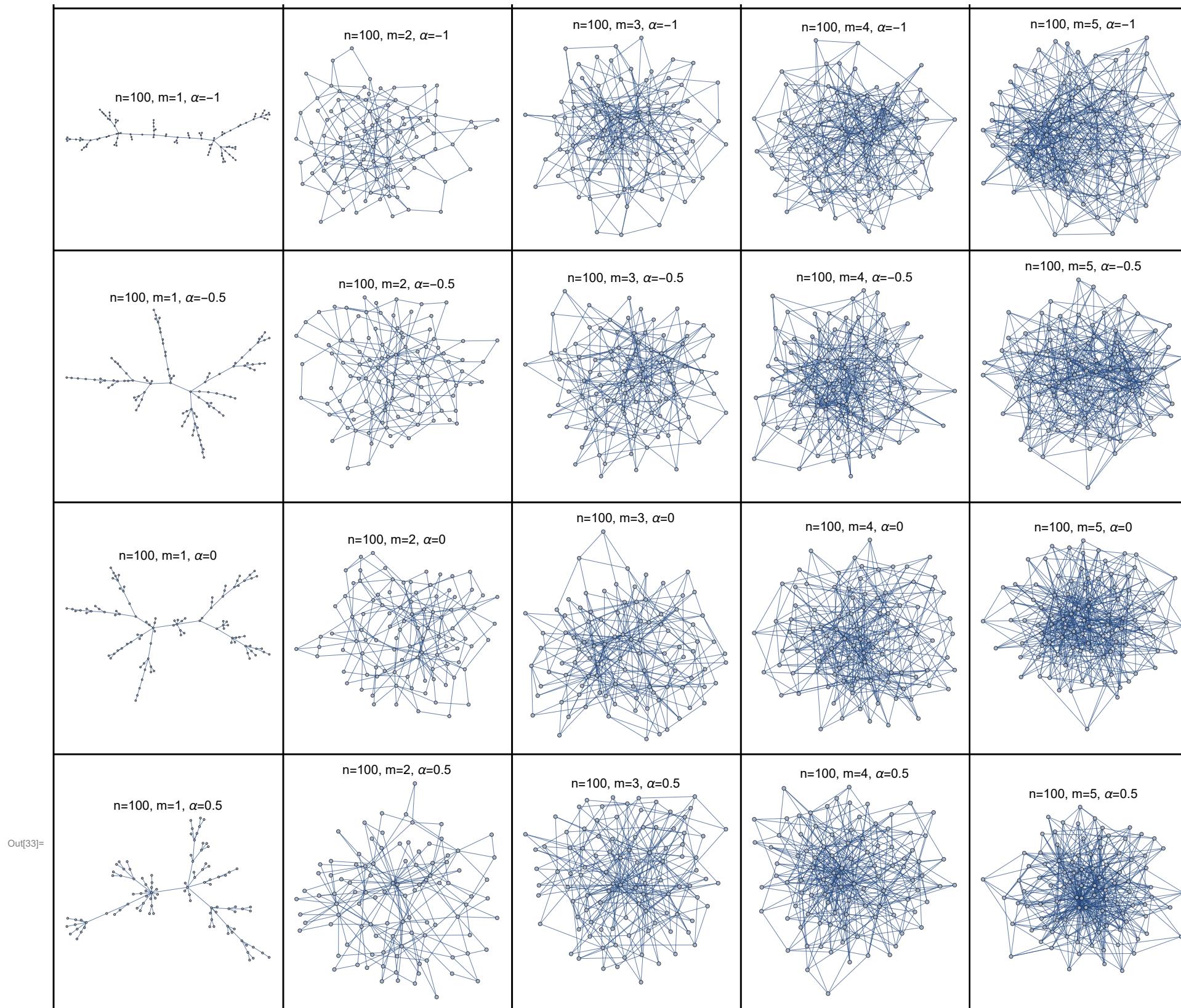


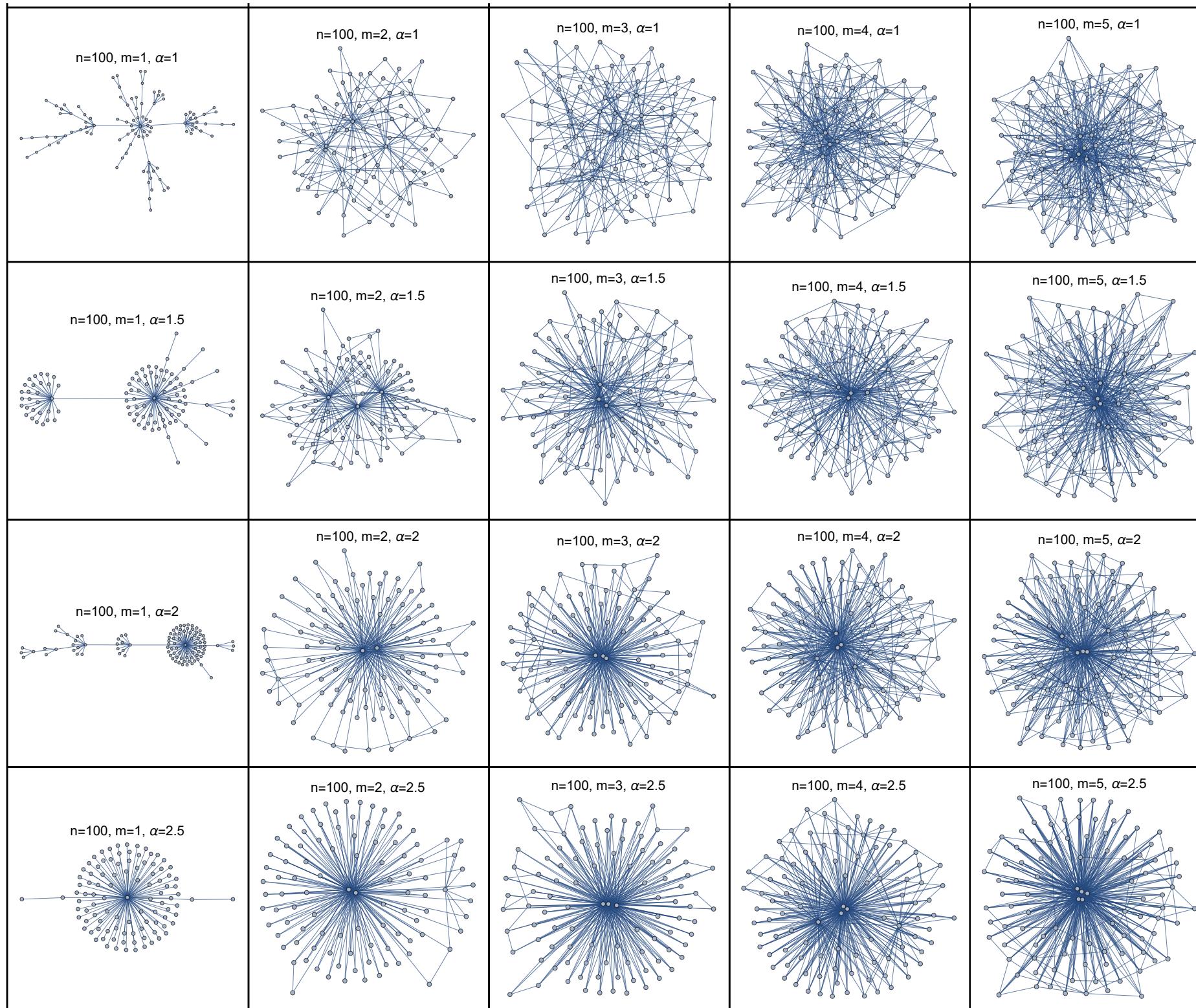
3. Модель Бьянкони-Барабаши

```
In[32]:= BianconiBarabasiGraphEdgeList[m_, α_, μ_] := Module[{n = Length[μ]}, If[n ≤ m, EdgeList[CompleteGraph[n]], (*m==1*) If[m == 1, Module[{el = PadRight[{1 → 2}, n - 1, {{}}], pl = {}, vl = Table[i, {i, 1, n}], vla = {}}, (*m==1 and α==0*) If[α == 0, Do[vla = RandomChoice[μ[[1 ;; i - 1]] → vl[[1 ;; i - 1]]; el[[i - 1]] = vla → i, {i, 3, n}]; el, (*m==1 and α==1*) If[α == 1, pl = μ; Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]; el[[i - 1]] = vla → i; pl[[vla]] = pl[[vla]] + μ[[vla]], {i, 3, n}]; el, (*m>1 and α≠0 and α≠1*) pl = μ; Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]; el[[i - 1]] = vla → i; pl[[vla]] = μ[[vla]] ( (pl[[vla]])^(1/α) + 1)^α, {i, 3, n}]; el]], (*m>1*) Module[{el = PadRight[EdgeList[CompleteGraph[m]], m n - 1/2 m (m + 1), {{}}], k = m (m - 1)/2, pl = {}, vl = Table[i, {i, 1, n}], vla = {}}, (*m>1 and α==0*) If[α == 0, Do[vla = RandomSample[μ[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]; Do[k++; el[[k]] = vla[[j]] → i, {j, 1, m}], {i, m + 1, n}]; el, (*m>1 and α==1*) If[α == 1, pl = Table[If[i ≤ m, μ[[i]] (m - 1), μ[[i]] m], {i, 1, n}]; Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]]; Do[k++; el[[k]] = vla[[j]] → i, {j, 1, m}], {i, m + 1, n}]; el, (*m>1 and α≠0 and α≠1*) pl = Table[If[i ≤ m, μ[[i]] (m - 1)^α, μ[[i]] m^α], {i, 1, n}]; Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]]; Do[k++; el[[k]] = vla[[j]] → i, {j, 1, m}], {i, m + 1, n}]; el, pl[[vla[[j]]]] = pl[[vla[[j]]]] + μ[[vla[[j]]]], {j, 1, m}], {i, m + 1, n}]; el], (*m>1*) Grid[Prepend[Table[Graph[BianconiBarabasiGraphEdgeList[p1, p2, RandomReal[{}, 100]], PlotLabel → "n=100, m=" <> ToString[p1] <> ", α=" <> ToString[If[p2 ∈ Integers, p2, N[p2]]], ImageSize → Small], {p2, -3/2, 5/2, 1/2}, {p1, 1, 5, 1}], {Style["Граф построенный по модели Бьянкони-Барабаши с заданными параметрами", 16], SpanFromLeft}], Dividers → All, Spacings → {1, 1}]]]
```

In[33]:= Grid[Prepend[Table[Graph[BianconiBarabasiGraphEdgeList[p1, p2, RandomReal[{}, 100]], PlotLabel → "n=100, m=" <> ToString[p1] <> ", α=" <> ToString[If[p2 ∈ Integers, p2, N[p2]]], ImageSize → Small], {p2, -3/2, 5/2, 1/2}, {p1, 1, 5, 1}], {Style["Граф построенный по модели Бьянкони-Барабаши с заданными параметрами", 16], SpanFromLeft}], Dividers → All, Spacings → {1, 1}]







Генерация эволюции роста степени первых m вершин в модели Бьянкони-Барабаши с заданными параметрами

```

In[34]:= BianconiBarabasiGraphVertexDegreeEvolution[m_, α_, μ_] := Module[{n = Length[μ]}, If[n ≤ m, PadRight[Table[VertexDegree[CompleteGraph[i]], {i, 1, n}]]^T,
  (*m==1*) If[m == 1, Module[{vd = PadRight[{0, 1}, n, {}], pl = {}, vl = Table[i, {i, 1, n}], vla = {}},
    (*m==1 and α==0*) If[α == 0, Do[vla = RandomChoice[μ[[1 ;; i - 1]] → vl[[1 ;; i - 1]]; If[vla == 1, vd[[i]] = vd[[i - 1]] + 1, vd[[i]] = vd[[i - 1]]], {i, 3, n}]; vd,
      (*m==1 and α==1*) If[α == 1, pl = μ;
        Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]];
          If[vla == 1, vd[[i]] = vd[[i - 1]] + 1, vd[[i]] = vd[[i - 1]]];
          pl[[vla]] = pl[[vla]] + μ[[vla]], {i, 3, n}]; vd,
        (*m>1 and α≠0 and α≠1*) pl = μ;
        Do[vla = RandomChoice[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]]];
          If[vla == 1, vd[[i]] = vd[[i - 1]] + 1, vd[[i]] = vd[[i - 1]]];
          pl[[vla]] = μ[[vla]]  $\left(\left(\frac{pl[[vla]]}{\mu[[vla]]}\right)^{\frac{1}{\alpha}} + 1\right)^{\alpha}$ , {i, 3, n}]; vd]];
      (*m>1*) Module[{vd = PadRight[PadRight[Table[VertexDegree[CompleteGraph[i]], {i, 1, m}]], n, {ConstantArray[{}, m]}]^T, pl = {}, vl = Table[i, {i, 1, n}], vla = {}},
        (*m>1 and α==0*) If[α == 0, pl = PadRight[Table[m - 1, {i, 1, m}], n, m];
          Do[vla = Sort[RandomSample[μ[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];
            Do[pl[[vla[[j]]]] = pl[[vla[[j]]]] + 1;
              vd[[j, i]] = pl[[j]], {j, 1, m}], {i, m + 1, n}]; vd,
        (*m>1 and α==1*) If[α == 1, pl = Table[If[i ≤ m, μ[[i]]^(m - 1), μ[[i]] m], {i, 1, n}];
          Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];
            Do[pl[[vla[[j]]]] = pl[[vla[[j]]]] + μ[[vla[[j]]]];
              vd[[j, i]] =  $\frac{pl[[j]]}{\mu[[j]]}$ , {j, 1, m}], {i, m + 1, n}]; vd,
        (*m>1 and α≠0 and α≠1*) pl = Table[If[i ≤ m, μ[[i]]^(m - 1)^\alpha, μ[[i]] m^\alpha], {i, 1, n}];
          Do[vla = Sort[RandomSample[pl[[1 ;; i - 1]] → vl[[1 ;; i - 1]], m]];
            Do[pl[[vla[[j]]]] = μ[[vla[[j]]]]  $\left(\left(\frac{pl[[vla[[j]]]}{\mu[[vla[[j]]]}\right)^{\frac{1}{\alpha}} + 1\right)^{\alpha}$ ;
              vd[[j, i]] =  $\left(\frac{pl[[j]]}{\mu[[j]]}\right)^{\frac{1}{\alpha}}$ , {j, 1, m}], {i, m + 1, n}]; vd];
      vd]]]]]

```

Покажем что рост степени вершины в модели Бьянкони-Барабаши подчиняется закону $k(t) = m \left(\frac{t}{t_i} \right)^{\frac{\mu_i}{C}}$, где при равномерном распределении $C \approx 1.255$

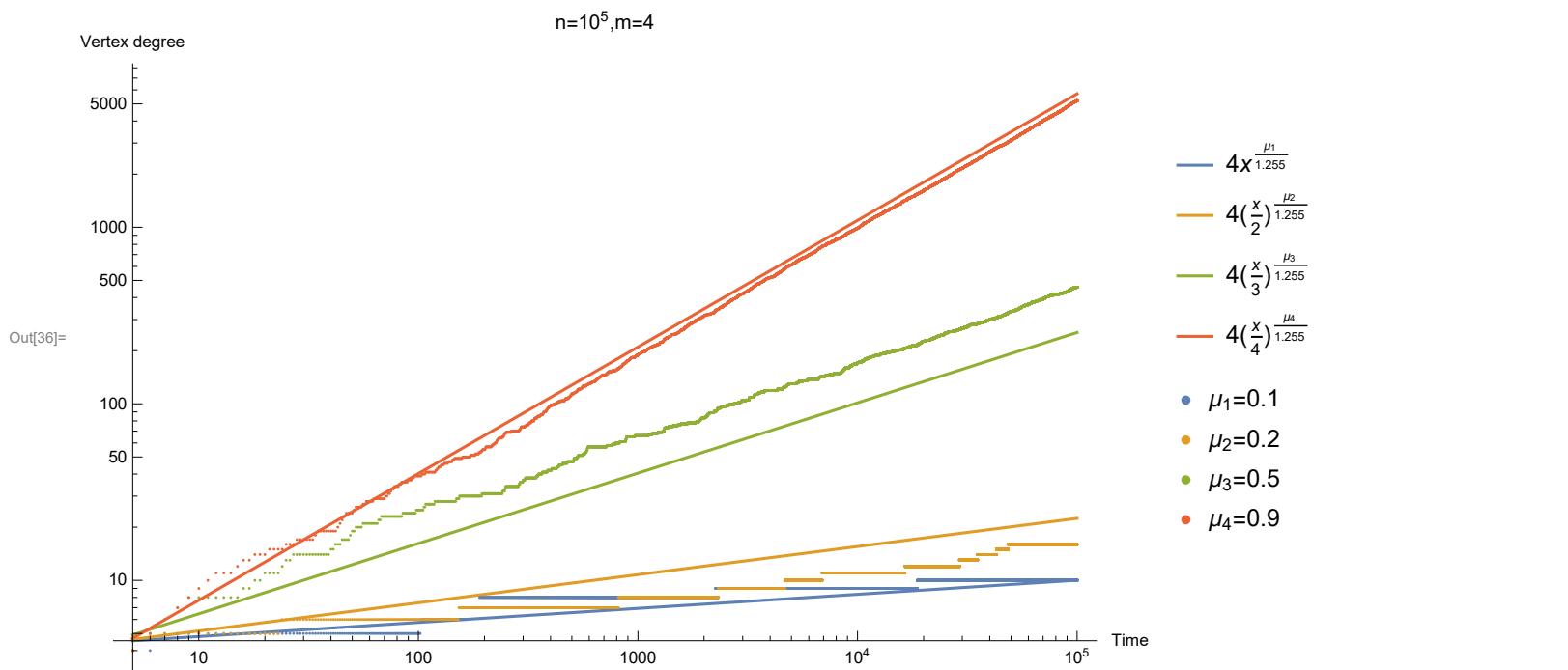
```
In[35]:= bbyde = Import[NotebookDirectory[] <> "\\sim2.m"]; (*bbyde=RandomVariate[UniformDistribution[],100000];*)
          импорт директория файла блокнота
          реализация слу... равномерное распределение

bbyde[[1]]=0.1;
bbyde[[2]]=0.2;
bbyde[[3]]=0.5;
bbyde[[4]]=0.9;
bbyde=BianconiBarabasiGraphVertexDegreeEvolution[4,1,bbyde];*)

In[36]:= Show[LogLogPlot[{4 x^(0.1/1.255), 4 (x/2)^(0.2/1.255), 4 (x/3)^(0.5/1.255), 4 (x/4)^(0.9/1.255)}, {x, 5, 100000}, PlotLegends -> {"4x^(μ₁/1.255)", "4(x/2)^(μ₂/1.255)", "4(x/3)^(μ₃/1.255)", "4(x/4)^(μ₄/1.255)"}],
          лог... график функции в лог-лог масштабе
          легенды графика

ListLogLogPlot[{Range[10^5], bbyde[[1]]}, {Range[10^5], bbyde[[2]]}, {Range[10^5], bbyde[[3]]}, {Range[10^5], bbyde[[4]]}], PlotLegends -> {"μ₁=0.1", "μ₂=0.2", "μ₃=0.5", "μ₄=0.9"}, 
          диаграмма разброса... диапазон
          диапазон
          диапазон
          легенды графика

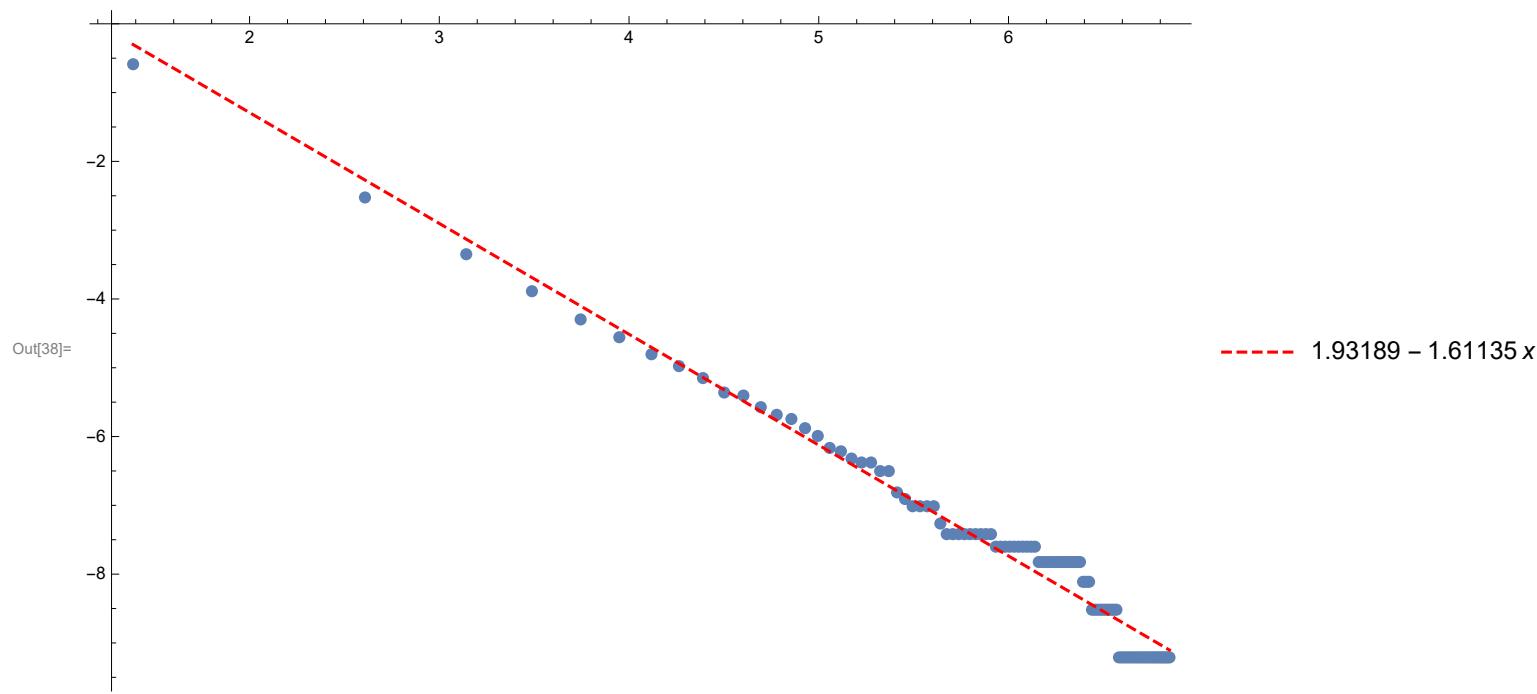
AxesLabel -> {"Time", "Vertex degree"}, PlotLabel -> "n=10^5,m=4", ImageSize -> Large]
          обозначения на осях
          пометка графика
          размер изоб... крупный
```



Посмотрим коэффициент степенного распределения

```
In[37]:= bbud = VertexDegree[Graph[BianconiBarabasiGraphEdgeList[4, 1, RandomVariate[UniformDistribution[], 10000]]]];
          степень верш... граф
          реализация слу... равномерное распределение
```

```
In[38]:= Module[{in = CCDF[bbud, 100], out = {}}, out = Log[in][[1 ;; Length[in] - 1]];
Show[ListPlot[out], Plot[Evaluate[Normal[LinearModelFit[out, x, x]]], {x, out[[1, 1]], out[[-1, 1]]}, PlotStyle -> {Red, Dashed}, PlotLegends -> "AllExpressions"], ImageSize -> Large]]
Out[38]=
```



Степенные распределения при различных распределениях μ

```
In[39]:= Module[{data = {ConstantArray[1, 10000], RandomVariate[UniformDistribution[], 10000],
 MinMaxNormalization[RandomVariate[NormalDistribution[], 10000]], MinMaxNormalization[RandomVariate[ExponentialDistribution[1], 10000]]}},
  LogLogPlot[Evaluate[Table[SurvivalFunction[EmpiricalDistribution[VertexDegree[Graph[BianconiBarabasiGraphEdgeList[4, 1, data[[i]]]]]], x], {i, 1, Length[data]}]],
  {x, 4, 10000}, Exclusions -> None, PlotLegends -> {"ConstantArray (BA model)", "UniformDistribution", "NormalDistribution", "ExponentialDistribution"},
  AxesLabel -> {"Vertex degree", "Probability"}, PlotLabel -> "CCDF", ImageSize -> Large]]
Out[39]=
```

