

# Physik der sozio-ökonomischen Systeme *mit dem Computer*

*JOHANN WOLFGANG GOETHE UNIVERSITÄT*  
*24.05.2024*

*MATTHIAS HANAUSKE*

*FRANKFURT INSTITUTE FOR ADVANCED STUDIES*  
*JOHANN WOLFGANG GOETHE UNIVERSITÄT*  
*INSTITUT FÜR THEORETISCHE PHYSIK*  
*ARBEITSGRUPPE RELATIVISTISCHE ASTROPHYSIK*  
*D-60438 FRANKFURT AM MAIN*  
*GERMANY*

## 6. Vorlesung

# Plan für die heutige Vorlesung

- Kurze Wiederholung der Inhalte der 5. Vorlesung
- Einführung in die Theorie der komplexen Netzwerke
  - Warum evolutionäre Spieltheorie auf komplexen Netzwerken?
  - Mathematische Beschreibung von komplexen Netzwerken
  - Definition wichtiger Netzwerk-charakterisierender Größen
  - Definition unterschiedlicher Netzwerk-Typen und Netzwerk-Klassen
  - Komplexe Netzwerke in der Realität
  - Die Klasse der zufälligen Netzwerke (random networks)
  - Die Klasse der „Kleine Welt“ Netzwerke (small world networks)
  - Komplexe Netzwerke analysieren mit Python und NetworkX

# Inhalte Vorlesung 5

- Numerisches Lösen von Differentialgleichungen
- Einführung in die evolutionäre Spieltheorie
  - Die Differentialgleichung eines evolutionären, symmetrischen  $(2 \times 3)$ -Spiels
    - Die 19 Zeeman – Klassen
- Anwendungsfelder Spieltheorie
  - Anwendungsfelder in den Wirtschafts- Sozialwissenschaften und Biologie
    - Experimentelle Ökonomie
    - Die Finanzkrise als Falke-Taube Spiel
    - Die Entstehung einer dritten Strategie im Elfmeter-Spiel (Nesken Effekt)
    - Evolutionäre Entwicklung einer Eidechsen Population als symmetrisches  $(2 \times 3)$ -Spiel
    - Das Räuber-Beute Spiel und die Lotka-Volterra-Gleichung

# Differentialgleichungen: Numerische Lösung von Anfangswertproblemen

Im vorigen Unterpunkt hatten wir die Bewegung einzelner Teilchen in einer Kiste simuliert. In der Physik ist die zeitliche Entwicklung eines Systems oft in Form von Differentialgleichungen (DGLs) gegeben. In diesem Unterpunkt betrachten wir das numerische Lösen einer Differentialgleichung erster Ordnung der Form

$$\dot{y}(t) = \frac{dy(t)}{dt} = f(t, y(t)) \quad , \text{ mit: } a \leq t \leq b, \quad y(a) = \alpha \quad .$$

Die Funktion  $f(t, y(t))$  bestimmt die DGL und somit das Verhalten der gesuchten Funktion  $y(t)$ . Es wird hierbei vorausgesetzt, dass  $f(t, y(t))$  auf einer Teilmenge  $\mathcal{D} = \{(t, y) | a \leq t \leq b, -\infty \leq y \leq \infty\}$  kontinuierlich definiert ist. Weiter wird angenommen, dass das so definierte Anfangswertproblem "well-posed" ist und eine eindeutige Lösung  $y(t)$  existiert ("well-posed" bedeutet hier, dass die Differentialgleichung eine Struktur hat, bei der kleine Störungen im Anfangszustand nicht exponentiell anwachsen). Wir hatten bereits gesehen, wie man Differentialgleichungen mittels Jupyter Notebooks und SymPy DGLs analytisch löst (siehe [Jupyter Notebooks und das Rechnen mit symbolischen Ausdrücken](#)). Nicht jede DGL lässt sich analytisch lösen und falls der Befehl "dsolve()" keine sinnvollen Resultate liefert, muss man die zeitliche Entwicklung der Funktion  $y(t)$  numerisch berechnen. Die numerische Lösung der DGL kann man sich auch direkt in Python mittels der Methode "integrate.odeint()" berechnen (Python-Modul "scipy" ). Möchte man die Lösung jedoch in einem C++ Programm berechnen, so ist man auf die Anwendung eines numerischen Verfahrens angewiesen.

# Numerische Verfahren zum Lösen von Differentialgleichung erster Ordnung

Das Euler Verfahren:

$$y_{i+1} = y_i + h \cdot f(t_i, y_i)$$

Mittelpunktmethode:

$$y_{i+1} = y_i + h \cdot \left[ f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} f(t_i, y_i)\right) \right]$$

Modifizierte Euler Methode:

$$y_{i+1} = y_i + \frac{h}{2} \cdot [f(t_i, y_i) + f(t_{i+1}, y_i + h f(t_i, y_i))]$$

Runge-Kutta Ordnung vier:

$$y_{i+1} = y_i + \frac{1}{6} \cdot (k_1 + 2k_2 + 2k_3 + k_4) \quad , \text{ wobei:}$$

$$k_1 = h f(t_i, y_i)$$

$$k_2 = h f\left(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_1\right)$$

$$k_3 = h f\left(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_2\right)$$

$$k_4 = h f(t_{i+1}, y_i + k_3)$$

## Systeme von gekoppelten Differentialgleichungen ▶

Wir betrachten zunächst das numerische Lösen eines Systems von  $m$ -gekoppelten Differentialgleichungen (DGLs) erster Ordnung der Form

$$\begin{aligned}\dot{y}_1(t) &= \frac{dy_1}{dt} = f_1(t, y_1, y_2, \dots, y_m) \\ \dot{y}_2(t) &= \frac{dy_2}{dt} = f_2(t, y_1, y_2, \dots, y_m) \\ \dot{y}_3(t) &= \dots = \\ &\dots = \dots \\ \dot{y}_m(t) &= \frac{dy_m}{dt} = f_m(t, y_1, y_2, \dots, y_m) \quad ,\end{aligned}$$

wobei die zeitliche Entwicklung der Vektorfunktion  $\vec{y}(t) = (y_1(t), y_2(t), \dots, y_m(t))$  in den Grenzen  $a \leq t \leq b$  gesucht wird.

Die  $m$ -Funktionen  $f_i(t, y_1, y_2, \dots, y_m)$ ,  $i \in [1, 2, \dots, m]$  bestimmen das System der DGLs und somit das Verhalten der gesuchten Funktion  $\vec{y}(t)$ . Es wird hierbei vorausgesetzt, dass die Funktionen  $f_i(t, y_1, y_2, \dots, y_m)$  auf einer Teilmenge  $\mathcal{D} \subseteq \mathbb{R}^{m+1}$  kontinuierlich definiert sind und das so definierte Anfangswertproblem "well-posed" ist und eine eindeutige Lösung  $\vec{y}(t)$  existiert. Bei gegebener Anfangskonfiguration

$$y_1(a) = \alpha_1, y_2(a) = \alpha_2, \dots, y_m(a) = \alpha_m$$

ist es dann numerisch möglich das System von gekoppelten DGLs zu lösen.

## Beispiel: Numerische Lösung eines Systems von zwei gekoppelten Differentialgleichungen erster Ordnung

Wir betrachten speziell das folgende System bestehend aus zwei gekoppelten DGLs ( $m = 2$ ):

$$\begin{aligned}\dot{y}_1(t) &= \frac{dy_1}{dt} = 3y_1 + 2y_2 - (2t^2 + 1) \cdot e^{2t} =: f_1(t, y_1, y_2) \\ \dot{y}_2(t) &= \frac{dy_2}{dt} = 4y_1 + y_2 + (t^2 + 2t - 4) \cdot e^{2t} =: f_2(t, y_1, y_2) \quad ,\end{aligned}$$

und sind an der numerischen Lösung  $\vec{y}(t) = (y_1(t), y_2(t))$  im Zeitintervall  $t \in [0, 1]$  interessiert. Die Anfangsbedingungen lauten

$$y_1(0) = \alpha_1 = 1, \quad y_2(0) = \alpha_2 = 1 \quad .$$

Das Lösen dieses Systems von DGLs ist auf gleichem Wege möglich, wie man einzelne Differentialgleichungen numerisch approximiert.

# Replikatorodynamik (2xM)-Spiele

Wir beschränken uns zunächst auf symmetrische (2xM)-Spiele, d.h. zwei Personen - M Strategien Spiele. Da es sich um symmetrische Spiele handelt, sind alle Spieler gleichberechtigt und man kann von einer homogenen Population ausgehen. Die Differentialgleichung der Replikatorodynamik beschreibt wie sich die einzelnen Populationsanteile der zur Zeit t gewählten Strategien  $x_j(t)$ ,  $j=1,2,\dots,M$  im Laufe der Zeit entwickeln.

$$\dot{x}_j(t) := \frac{dx_j(t)}{dt} = x_j(t) \cdot \left[ \underbrace{\sum_{k=1}^M \$_{jk} \cdot x_k(t)}_{\text{Fitness der Strategie } j} - \underbrace{\sum_{l=1}^M \sum_{k=1}^M \$_{kl} \cdot x_k(t) \cdot x_l(t)}_{\text{Durschnittliche Fitness (Auszahlung) der gesamten Population}} \right]$$

Wobei die Parameter  $\$_{kl}$  die einzelnen Einträge in der Auszahlungsmatrix des 1. Spielers darstellen

$$\hat{\$} = \hat{\$}^1 = \begin{pmatrix} \$_{11} & \$_{12} & \$_{13} & \dots & \$_{1M} \\ \$_{21} & \$_{22} & \$_{23} & \dots & \$_{2M} \\ \$_{31} & \$_{32} & \$_{33} & \dots & \$_{3M} \\ \dots & \dots & \dots & \dots & \dots \\ \$_{M1} & \$_{M2} & \$_{M3} & \dots & \$_{MM} \end{pmatrix}$$

**Fitness der Strategie j**

Durchschnittlicher Erfolg der j-ten Strategie

**Durschnittliche Fitness (Auszahlung) der gesamten Population**



# Replikatorodynamik

(für symmetrische (2x3)-Spiele)

Man erhält ein System von drei gekoppelten Differentialgleichungen:

$$\frac{dx_1}{dt} = x_1 \cdot [\$_{11} \cdot x_1 + \$_{12} \cdot x_2 + \$_{13} \cdot x_3 - \bar{\$}]$$

$$\frac{dx_2}{dt} = x_2 \cdot [\$_{21} \cdot x_1 + \$_{22} \cdot x_2 + \$_{23} \cdot x_3 - \bar{\$}]$$

$$\frac{dx_3}{dt} = x_3 \cdot [\$_{31} \cdot x_1 + \$_{32} \cdot x_2 + \$_{33} \cdot x_3 - \bar{\$}]$$

Das System von Differentialgleichungen lässt sich bei gegebener Auszahlungsmatrix  $\hat{\$}$  und Anfangsbedingung  $(x_1(0), x_2(0), x_3(0))$  meist nur numerisch (auf dem Computer) lösen. Die Lösungen bestehen dann aus den drei (zeitlich abhängigen) Populationsanteilen  $(x_1(t), x_2(t), x_3(t))$ .

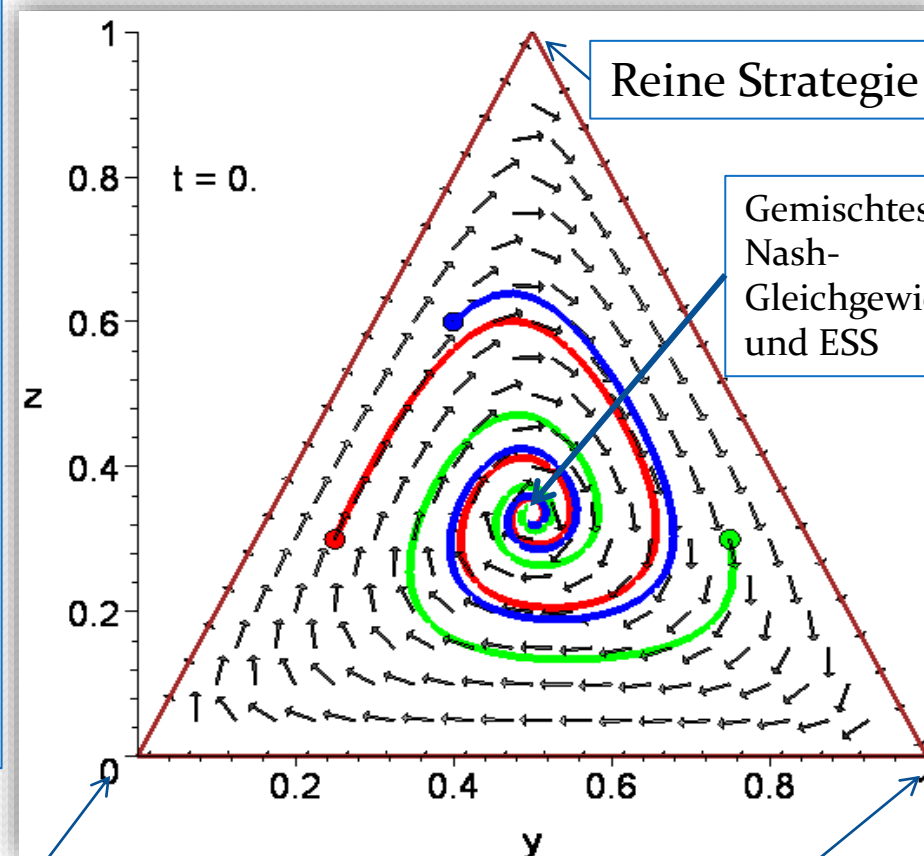
# Replikatorodynamik

(für symmetrische (2x3)-Spiele, **Beispiel 1**)

Wir betrachten im Folgenden ein Beispiel eines (2x3)-Spiels mit der rechts angegebenen Auszahlungsstruktur:

	Strategie 1	Strategie 2	Strategie 3
Strategie 1	(0, 0)	(2, -1)	(-1, 2)
Strategie 2	(-1, 2)	(0, 0)	(2, -1)
Strategie 3	(2, -1)	(-1, 2)	(0, 0)

Die rechte Abbildung zeigt die zeitliche Entwicklung der relativen Populationsanteile der gewählten Strategien für drei mögliche Anfangsbedingungen. Die einzige evolutionär stabile Strategie dieses Beispiels befindet sich beim gemischten Nash-Gleichgewicht  $\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ . Die einzelnen Pfeile im Dreieck veranschaulichen den durch die Spielmatrix bestimmten Strategien-„Richtungswind“, dem die Population zeitlich folgen wird.



Zur Visualisierung der evolutionären Entwicklung benutzt man oft die sogen. barycentric coordinates:

$$y := x_2 + \frac{x_3}{2}$$

$$z := x_3$$

Reine Strategie 1

Reine Strategie 2

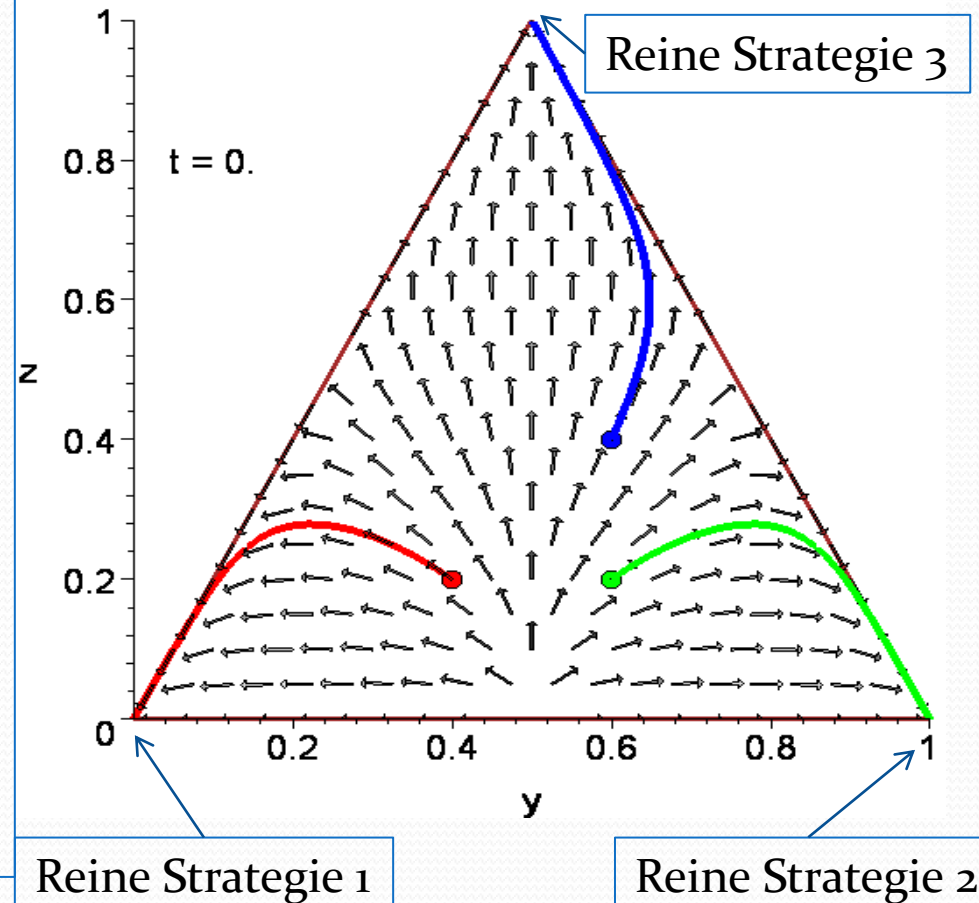
# Replikatorodynamik

(für symmetrische (2x3)-Spiele, **Beispiel 2**)

Wir betrachten im Folgenden ein Beispiel eines (2x3)-Spiels mit der rechts angegebenen Auszahlungsstruktur:

	Strategie 1	Strategie 2	Strategie 3
Strategie 1	(0, 0)	(-3, -3)	(-1, -1)
Strategie 2	(-3, -3)	(0, 0)	(-1, -1)
Strategie 3	(-1, -1)	(-1, -1)	(0, 0)

Die rechte Abbildung zeigt die zeitliche Entwicklung der relativen Populationsanteile der gewählten Strategien für drei mögliche Anfangsbedingungen. Das Spiel besitzt drei Nash-Gleichgewichte in reinen Strategien, die ebenfalls evolutionär stabile Strategien darstellen. Welche der drei ESS die Population realisiert hängt von dem Anfangswert der Populationsanteile ab. Die zeitliche Entwicklung folgt wieder dem Strategien-„Richtungswind“ der zugrundeliegenden Auszahlungsmatrix.

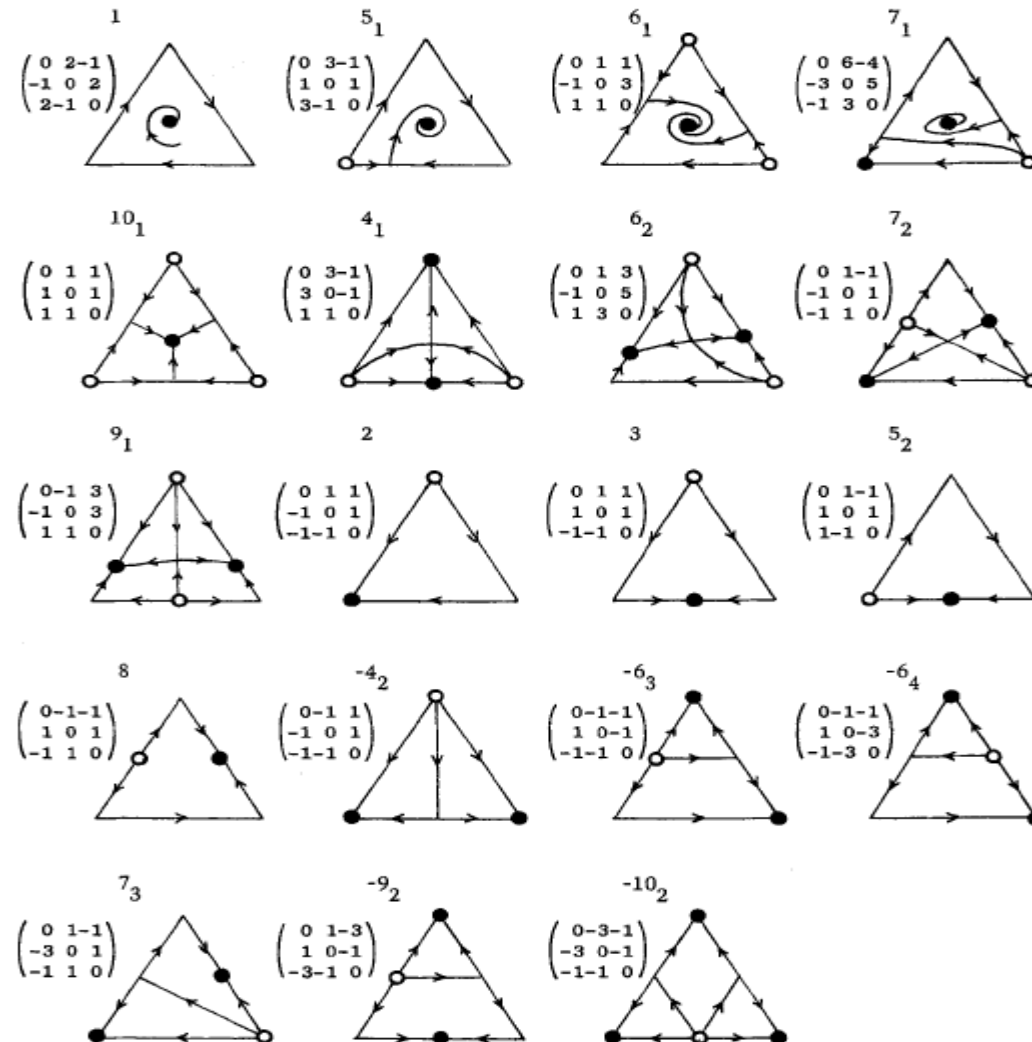


# Replikatorodynamik

(Klassifizierung symmetrische (2x3)-Spiele)

E. C. Zeeman, *POPULATION DYNAMICS FROM GAME THEORY*,  
In: *Global Theory of Dynamical Systems*, Springer 1980

E. C. Zeeman zeigt in seinem im Jahre 1980 veröffentlichten Artikel, dass man evolutionäre, symmetrische (2x3)-Spiele in 19 Klassen einteilen kann. Die Abbildung rechts zeigt das evolutionäre Verhalten dieser 19 Spieltypen. Die ausgefüllten schwarzen Punkte markieren die evolutionär stabilen Strategien der jeweiligen Spiele. Es gibt Spielklassen, die besitzen lediglich eine ESS und Klassen die sogar drei ESS besitzen.



# Physik der sozio-ökonomischen Systeme mit dem Computer

(Physics of Socio-Economic Systems with the Computer)

Vorlesung gehalten an der J.W.Goethe-Universität in Frankfurt am Main  
(Sommersemester 2024)

von Dr.phil.nat. Dr.rer.pol. Matthias Hanauke

Frankfurt am Main 01.02.2024

Erster Vorlesungsteil:

Die 19 Klassen der evolutionären symmetrischen (2 x 3)-Spiele

## Einführung

In diesem Unterkapitel werden die evolutionären symmetrischen (2 x 3)-Spiele analysiert. Symmetrische (2 x m)-Spiele werden

Aufgrund der Symmetrie der Auszahlungsmatrix  $\hat{s} = \hat{s}^T$  kann die zeitliche Entwicklung solcher Spiele durch nur einen Populationsvektor  $\vec{x}(t) = (x_1(t), x_2(t), \dots, x_m(t))$

Wir beschränken uns im Folgenden auf den Fall  $m = 3$  und setzen die Auszahlungsmatrix wie folgt an:

$$\hat{s} = \begin{pmatrix} s_{11} & s_{12} & s_{13} \\ s_{21} & s_{22} & s_{23} \\ s_{31} & s_{32} & s_{33} \end{pmatrix}$$

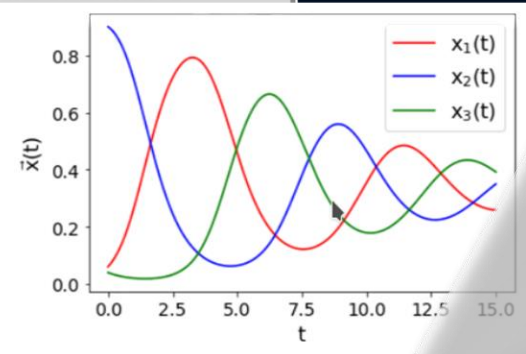
$$z_2(t) = 1 - x_1(t) - x_2(t)$$

Aufgrund der Normalisierungsbedingung des Populationsvektors kann man den Anteil der Population, die die Strategie drei spielt, durch die anderen beiden Anteile ausdrücken:  $z_3(t) = 1 - x_1(t) - x_2(t)$

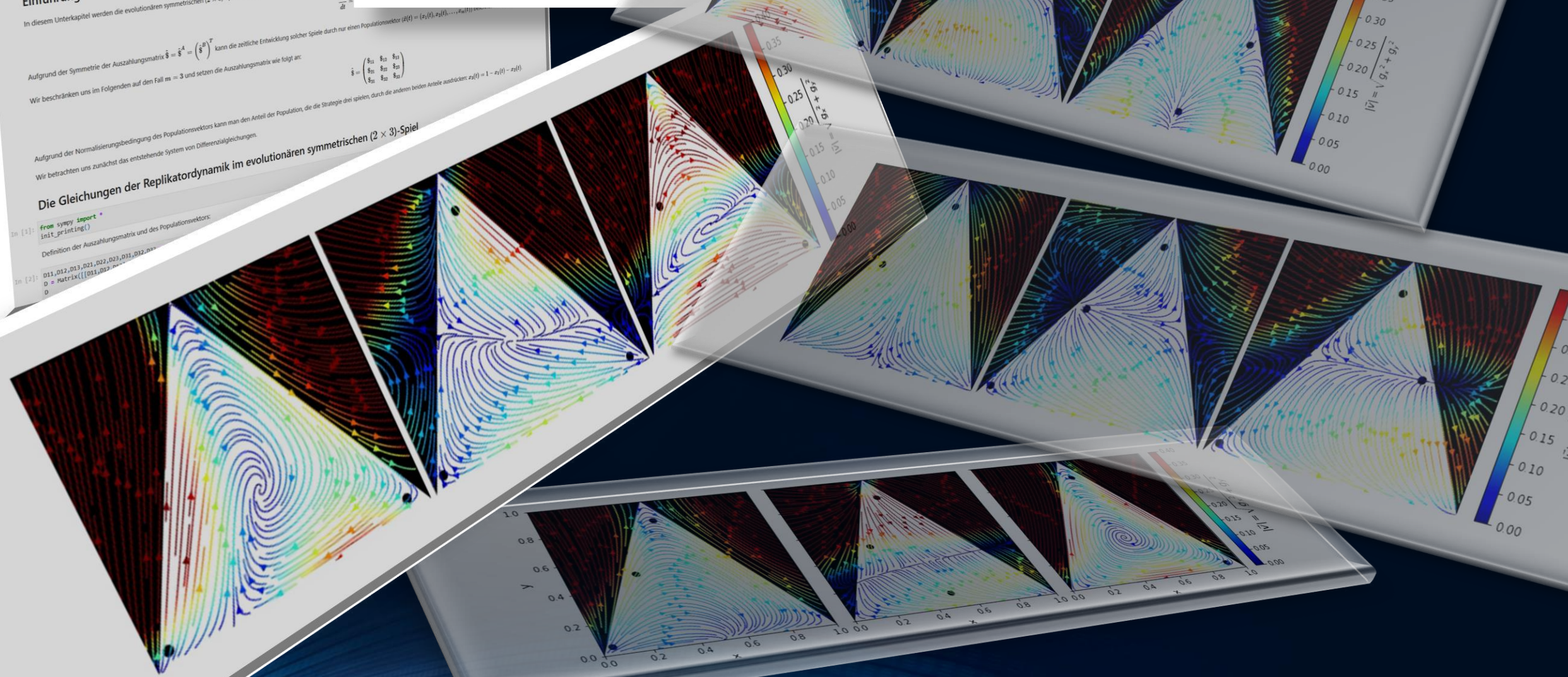
Wir betrachten uns zunächst das entstehende System von Differenzialgleichungen.

## Die Gleichungen der Replikatorndynamik im evolutionären symmetrischen (2 x 3)-Spiel

```
In [1]: from sympy import *\ninit_printing()\n\nDefinition der Auszahlungsmatrix und des Populationsvektors:\n\nIn [2]: D11, D12, D13, D21, D22, D23, D31, D32, D33 = symbols('D11 D12 D13 D21 D22 D23 D31 D32 D33')\nD = Matrix([[D11, D12, D13], [D21, D22, D23], [D31, D32, D33]])
```



# Jupyter Notebook Evolutionenspiel4.ipynb



## Anwendungsfelder der Spieltheorie

Spieltheoretische Konzepte werden in den unterschiedlichsten Fachdisziplinen angewandt und reichen von den Wirtschaftswissenschaften, der Soziologie hin zur Biologie. Der Arbeitskreis des Fachverbandes der Deutsche Physikalische Gesellschaft (siehe DPG-Fachverband: Physik der sozio-ökonomischen Systeme) befasst sich mit diesem stark interdisziplinären Forschungsfeld und diverse Anwendungsfelder werden auf den jährlichen Fachtagungen diskutiert. Im Folgenden werden einige Beispiele von Anwendungsfeldern der Spieltheorie und weiterführende Literaturlinks aufgelistet.

### Anwendungsbeispiele im Bereich der Ökonomie

- Experimentelle Ökonomie am Beispiel des Gefangenendilemmas
- Vertrauen und Fairness in Allgemeingüter Spielen
- Kooperation und Fairness am Beispiel von Markt- und Ultimatum-Spielen
- Die Finanzkrise als Falke-Taube Spiel
- Evolutionäre Dynamik im Öffentliche-Güter-Spiel
- Spieltheorie und Auktionskonzepte (Nobelpreis für Wirtschaftswissenschaften 2020)

### Anwendungsbeispiele im Bereich der Biologie

- Das Buch von Martin A. Nowak Evolutionary Dynamics - Exploring the Equations of Life, 2006 diskutiert mehrere Anwendungsfelder der evolutionären Spieltheorie im Bereich der Biologie, wie z.B. die evolutionäre Dynamik von Krebs, HIV Infektion und die Evolution der Virulenz.
- Paarungsverhalten einer Eidechsenpopulation als evolutionäres symmetrisches (2x3)-Spiel
- Evolutionäre Entwicklung von Makro-Molekülen
- Das Gefangenendilemma eines RNA-Virus

### Anwendungsbeispiele im Bereich der Sozialwissenschaft

- Evolution von sozialen Normen
- Evolution der Moral
- Evolution der Sprachentwicklung einer Population
- Das Buch von Martin A. Nowak Evolutionary Dynamics - Exploring the Equations of Life, 2006 diskutiert auch am Ende die Evolution der Sprachentwicklung einer Population.
- Anwendungen in den Politikwissenschaften
- Die Entstehung einer dritten Strategie im Elfmeter-Spiel (Neeskens Effekt)

Anwendungsfelder  
Spieltheorie

# E-Learning und interaktive Übungsaufgaben

Zusätzlich zu den Informationen aus dieser Internetseite finden Sie in diesem Unterpunkt diverse interaktive Übungsaufgaben zu den folgenden Themen:

## Aufgabe 1

Reine Nash-Gleichgewichte in einem simultanen (2x2)-Spiel in strategischer Form mit symmetrischer Auszahlungsmatrix

## Aufgabe 2

Gemischtes Nash-Gleichgewicht in einem simultanen (2x2)-Spiel in strategischer Form mit symmetrischer Auszahlungsmatrix

## Aufgabe 3

Das gemischtes Nash-Gleichgewicht im Hirschjagt-Spiel

## Aufgabe 4

Spielklassen von simultanen (2x2)-Spielen in strategischer Form mit symmetrischer Auszahlungsmatrix

## Aufgabe 5

Zeitliche Entwicklung des Populationsvektors im evolutionären Spiel

## Aufgabe 6

Evolutionär stabile Strategien

## Aufgabe 7

Zeitliche Entwicklung des Populationsvektors im evolutionären Bi-Matrix Spiel

## Aufgabe 8

Gemischtes Nash-Gleichgewicht in einem simultanen (2x2)-Spiel in strategischer Form mit unsymmetrischer Auszahlungsmatrix

## Aufgabe 9

Gemischtes Nash-Gleichgewicht und zeitliche Entwicklung des Populationsvektors in Zentrumsspielen

## Aufgabe 10

Zeitliche Entwicklung des Populationsvektors im evolutionären (2x3)-Spiel

## Aufgabe 11

Gemischtes Nash-Gleichgewicht im evolutionären (2x3)-Spiel

## Aufgabe 12

Mittlere Distanz zwischen zwei Knoten in einem zufälligen Netzwerk

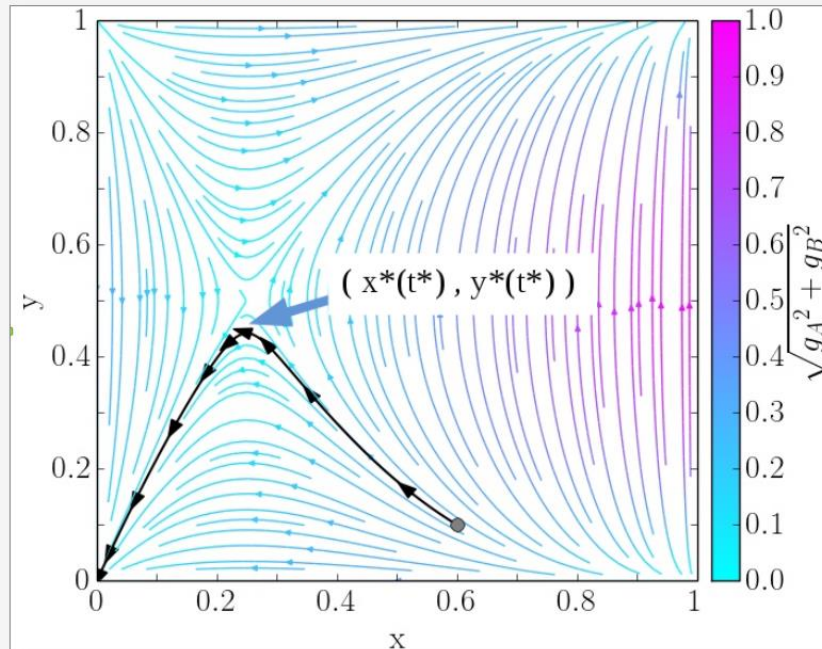
# Aufgabe 7

Das zeitliche Verhalten der Komponenten der Populationsvektoren (Gruppe A:  $x(t) := x_1^A(t)$  und Gruppe B:  $y(t) := x_1^B(t)$ ) wird in der Reproduktionsdynamik mittels des folgenden Systems von Differentialgleichungen beschrieben:

$$\frac{dx(t)}{dt} = [(\$_{11}^A + \$_{22}^A - \$_{12}^A - \$_{21}^A) y(t) + (\$_{12}^A - \$_{22}^A)] (x(t) - (x(t))^2) =: g_A(x, y)$$

$$\frac{dy(t)}{dt} = [(\$_{11}^B + \$_{22}^B - \$_{12}^B - \$_{21}^B) x(t) + (\$_{12}^B - \$_{22}^B)] (y(t) - (y(t))^2) =: g_B(x, y)$$

Qualitative Veranschaulichung der Aufgabenstellung



## Lösung

Der maximale Wert  $y$ -Wert beträgt  $y^* =$   
und er wird zum Zeitpunkt  $t^* =$  erreicht.

[Weiter zur nächsten Aufgabe ...](#)

Das durch die folgende Auszahlungstabelle definierte Bimatrix Spiel gehört der Klasse der Sattelpunktsspiele an.

A/B	$s_1$	$s_2$
$s_1$	( 10 , 10 )	( 4 , 7 )
$s_2$	( 9 , 4 )	( 5 , 5 )

Der Populationsvektor zur Zeit  $t=0$  sei  $(x(0) = 0.6, y(0) = 0.07696)$ . Der Anteil der Spieler in der Gruppe B die die Strategie  $s_1^B$  spielen nimmt zunächst zu, erreicht dann ein Maximum und nimmt dann wieder ab (siehe nebenstehende Abbildung). Berechnen Sie den Zeitpunkt  $t^*$  an dem der maximale Wert  $y^*$  erreicht wird. Tragen Sie bitte die beiden Werte in die unteren Eingabefelder ein

$t^* =$   ,  $y^* =$

und vergleichen Sie indem Sie den folgenden *Button* drücken.



## Aufgabe 8

A/B	$s_1$	$s_2$
$s_1$	( 366 , 246 )	( 120 , 7 )
$s_2$	( 112 , 141 )	( 215 , 354 )

Betrachten Sie die gemischte Erweiterung eines simultanen (2 Spieler)-(2 Strategien) Spiels in strategischer Form mit unsymmetrischer Auszahlungsmatrix. Die Menge der Spieler sei  $\mathcal{I} = \{A, B\}$ , die Menge der reinen Strategien sei  $\mathcal{S}^A = \mathcal{S}^B = \{s_1, s_2\}$  und die Präferenzordnungen der Spieler sei durch die neben stehende Auszahlungstabelle quantifiziert. Die reinen Strategien entsprechen den folgenden gemischten Strategien:  $s_1 \triangleq \tilde{s}^B = \tilde{s}^B = 1$  und  $s_2 \triangleq \tilde{s}^A = \tilde{s}^B = 0$ .

Bei welcher gemischten Strategienkombination  $(\tilde{s}^{A*}, \tilde{s}^{B*})$  befindet sich das gemischte Nash-Gleichgewicht? Tragen Sie bitte Ihre Werte in die unteren Eingabefelder ein

$\tilde{s}^{A*} =$   ,  $\tilde{s}^{B*} =$

und vergleichen Sie indem Sie den folgenden *Button* drücken.

## Lösung

Das gemischte Nash-Gleichgewicht besindet sich bei

$\tilde{s}^{A*} =$

$\tilde{s}^{B*} =$

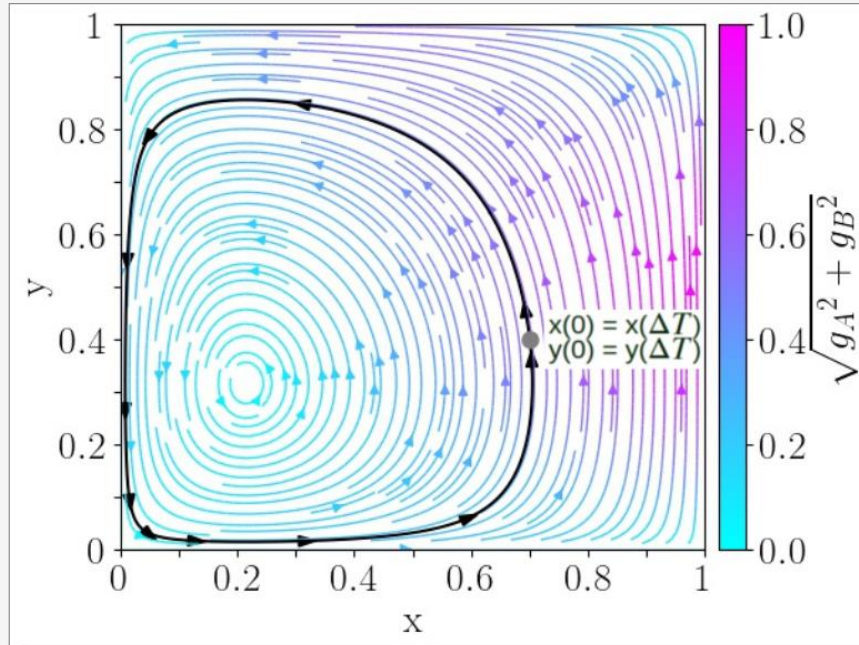
# Aufgabe 9

Das zeitliche Verhalten der Komponenten der Populationsvektoren (Gruppe A:  $x(t) := x_1^A(t)$  und Gruppe B:  $y(t) := x_1^B(t)$ ) wird in der Reproduktionsdynamik mittels des folgenden Systems von Differentialgleichungen beschrieben:

$$\frac{dx(t)}{dt} = [(\$_{11}^A + \$_{22}^A - \$_{12}^A - \$_{21}^A) y(t) + (\$_{12}^A - \$_{22}^A)] (x(t) - (x(t))^2) =: g_A(x, y)$$

$$\frac{dy(t)}{dt} = [(\$_{11}^B + \$_{22}^B - \$_{12}^B - \$_{21}^B) x(t) + (\$_{12}^B - \$_{22}^B)] (y(t) - (y(t))^2) =: g_B(x, y)$$

Qualitative Veranschaulichung der Aufgabenstellung



## Lösung

Die Werte der Auszahlungsmatrix betragen  $c_A =$  und  $c_B =$

und die Zeit eines Umlaufes des Populationsvektors beträgt  $\Delta T =$

Das durch die folgende Auszahlungstabelle definierte Bimatrix Spiel gehört der Klasse der Zentrumsspiele an.

A/B	$s_1^B$	$s_2^B$
$s_1^A$	( 8 , 8 )	( 6 , $c_B$ )
$s_2^A$	( $c_A$ , 4 )	( 5 , 5 )

Der Populationsvektor zur Zeit  $t=0$  sei  $(x(0)=0.7, y(0)=0.4)$ . Wählen Sie die noch offenen Werte der Auszahlungsmatrix ( $c_A$  und  $c_B$ ) so, dass sich das gemischte Nash-Gleichgewicht des Spiels bei  $x^* = \bar{s}^{A*} = 0.2593103$  und  $y^* = \bar{s}^{B*} = 0.3593103$  befindet. Berechnen Sie dann die Zeit  $\Delta T$ , die die Population benötigt um wieder zu ihrer Anfangskonfiguration zurück zu kehren:  $x(0) = x(\Delta T)$  und  $y(0) = y(\Delta T)$ . Tragen Sie bitte die berechneten Werte in die unteren Eingabefelder ein

$c_A =$   ,  $c_B =$

$\Delta T =$

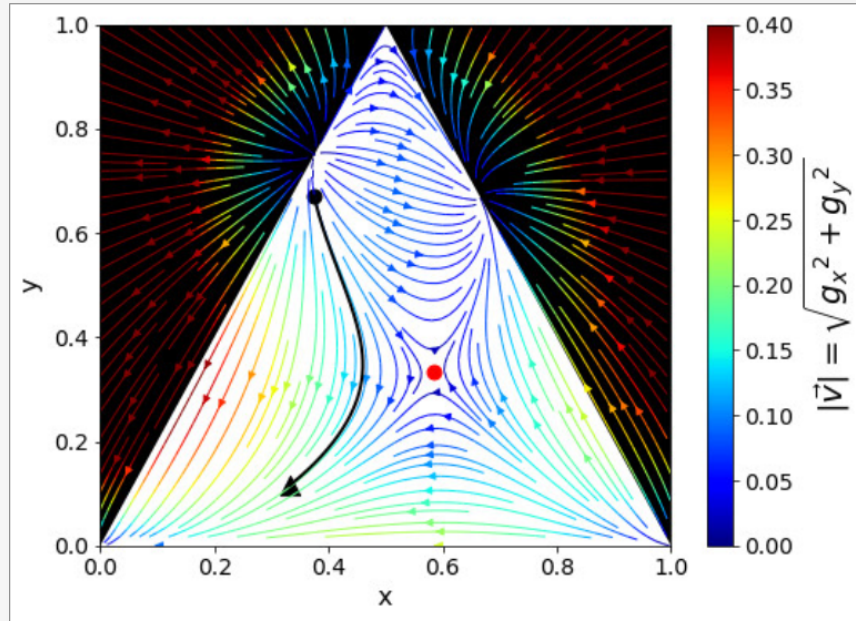
und vergleichen Sie indem Sie den folgenden *Button* drücken.

# Aufgabe 10

Betrachten Sie die zeitliche Entwicklung eines evolutionären (2 Spieler)-3 Strategien Spiels mit symmetrischer Auszahlungsmatrix. Das zeitliche Verhalten der Komponenten des Populationsvektors  $\vec{x}(t) = (x_1(t), x_2(t), x_3(t))$  wird in der Reproduktionsdynamik mittels des folgenden Systems von Differentialgleichungen beschrieben:

$$\frac{dx_1(t)}{dt} = g_1(x_1, x_2, x_3), \quad \frac{dx_2(t)}{dt} = g_2(x_1, x_2, x_3), \quad \frac{dx_3(t)}{dt} = g_3(x_1, x_2, x_3)$$

Qualitative Veranschaulichung der Aufgabenstellung



## Lösung

Der Wert des Parameters  $d$  in der Auszahlungsmatrix beträgt  $d =$

und die Position des Populationsvektors nach  $t = 2$  Zeiteinheiten ist  $x(t = 2) =$ ,  $y(t = 2) =$

[Weiter zur nächsten Aufgabe...](#)

Die Präferenzordnungen der Spieler sei durch die untere Auszahlungstabelle quantifiziert.

A/B	$s_1^B$	$s_2^B$	$s_3^B$
$s_1^A$	( 0 , 0 )	( 1 , -1 )	( -1 , -d )
$s_2^A$	( -1 , 1 )	( 0 , 0 )	( 1 , 2 )
$s_3^A$	( -d , -1 )	( 2 , 1 )	( 0 , 0 )

Aufgrund der Normalisierungsbedingung des Populationsvektors kann man sich die zeitliche Entwicklung der Strategiewahl der Population in einem baryzentrischen Dreiecks-Koordinatensystem veranschaulichen, wobei der x-Achsen Wert durch  $x := x_2 + x_3/2$  und der y-Achsen Wert durch  $y := x_3$  definiert ist. Die nebenstehende Abbildung veranschaulicht eine solche Entwicklung im baryzentrischen Koordinatensystem. Der rote Punkt innerhalb des Dreiecks ist dadurch gekennzeichnet, dass in ihm der Betrag des "Populationswindes" ( $|\vec{v}| = \sqrt{g_1^2 + g_2^2 + g_3^2}$ ) gerade Null ist. Wählen Sie den freien Parameter  $d$  der Auszahlungstabelle so, dass der rote Punkt bei  $(x_{Rot} = 0.6067708, y_{Rot} = 0.3333333)$  liegt. Nehmen Sie dann an, dass die Anfangs-Strategiewahl der Population (schwarzer Punkt in der nebenstehende Abbildung) bei  $(x(t = 0) = 0.375, y(t = 0) = 0.67)$  liegt und berechnen Sie die Position nach  $t = 2$  Zeiteinheiten (schwarzes Dreieck in der nebenstehende Abbildung). Tragen Sie bitte die berechneten Werte in die unteren Eingabefelder ein

$d =$

$x(t = 2) =$    $y(t = 2) =$

und vergleichen Sie indem Sie den folgenden *Button* drücken.

# Aufgabe 11

A/B	$s_1^B$	$s_2^B$	$s_3^B$
$s_1^A$	( 0 , 0 )	( d , -1 )	( -1 , 2 )
$s_2^A$	( -1 , d )	( 0 , 0 )	( 2 , -1 )
$s_3^A$	( 2 , -1 )	( -1 , 2 )	( 0 , 0 )

## Lösung

Das gemischte Nash-Gleichgewicht befindet sich bei

$$\begin{aligned} \tilde{s}_1^* &= , \tilde{s}_2^* = \\ \tilde{s}_3^* &= \\ x^* &= , y^* = \end{aligned}$$

Ihre Simulation des evolutionären Spiels sollte mit folgendem Bild übereinstimmen:

Betrachten Sie die zeitliche Entwicklung eines evolutionären (2 Spieler)-(3 Strategien) Spiels mit symmetrischer Auszahlungsmatrix. Das zeitliche Verhalten der Komponenten des Populationsvektors  $\vec{x}(t) = (x_1(t), x_2(t), x_3(t))$  wird in der Reproduktionsdynamik mittels des folgenden Systems von Differentialgleichungen beschrieben:

$$\frac{dx_1(t)}{dt} = g_1(x_1, x_2, x_3), \quad \frac{dx_2(t)}{dt} = g_2(x_1, x_2, x_3), \quad \frac{dx_3(t)}{dt} = g_3(x_1, x_2, x_3)$$

Die Präferenzordnungen der Spieler sei durch die neben stehende Auszahlungstabelle quantifiziert, wobei der Parameter d auf den folgenden Wert festgelegt sei  $d = 2.34949495$ . Aufgrund der Normalisierungsbedingung des Populationsvektors kann man sich die zeitliche Entwicklung der Strategiewahl der Population in einem baryzentrischen Dreiecks-Koordinatensystem veranschaulichen, wobei der x-Achsen Wert durch  $x := x_2 + x_3/2$  und der y-Achsen Wert durch  $y := x_3$  definiert ist. Bei welcher gemischten Strategienkombination  $(\tilde{s}_1^*, \tilde{s}_2^*, \tilde{s}_3^*)$  befindet sich das gemischte Nash-Gleichgewicht und die evolutionär stabile Strategie des Spiels? Tragen Sie bitte Ihre Werte in die unteren Eingabefelder ein und geben Sie zusätzlich noch die baryzentrischen Koordinaten  $(x^*, y^*)$  des gemischten Nash-Gleichgewichts an.

$$\begin{aligned} \tilde{s}_1^* &= \text{0,000} , \tilde{s}_2^* = \text{0,000} \\ \tilde{s}_3^* &= \text{0,000} \\ x^* &= \text{0,000} , y^* = \text{0,000} \end{aligned}$$

Überprüfen Sie ihre Berechnungen, indem Sie das evolutionäre Spiel simulieren. Wählen Sie dabei den Anfangswert der Strategiewahl der Population zu  $(x_0 = 0.92, y_0 = 0.04)$  (baryzentrisches Dreiecks-Koordinatensystem). Vergleichen Sie Ihre Ergebnisse, indem Sie den folgenden *Button* drücken.

Lösung anzeigen

# Einführung in die Theorie der komplexen Netzwerke

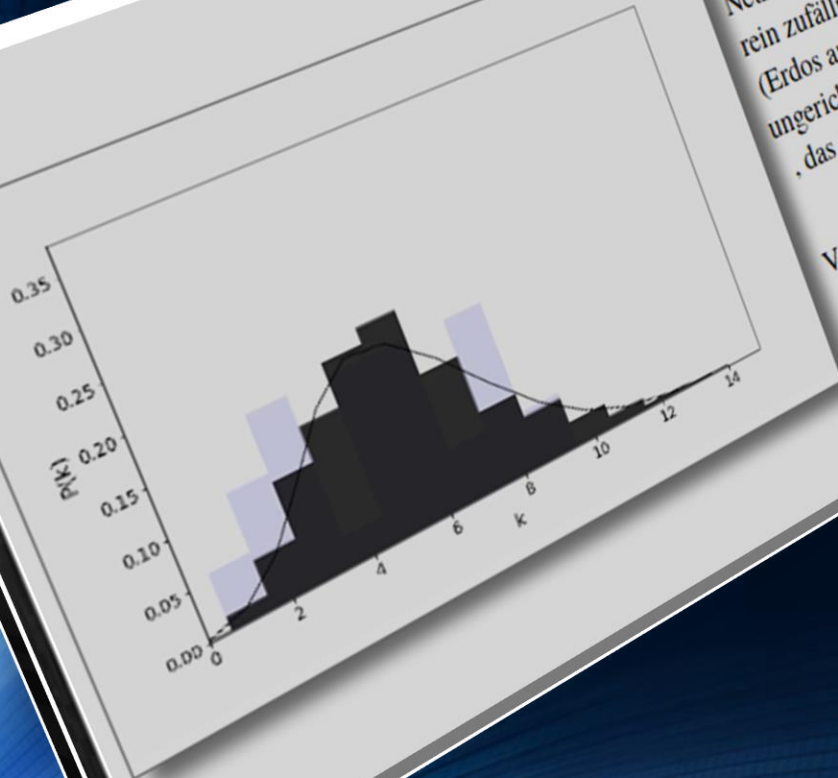
Internetseite der Vorlesung

## Zufällige Netzwerke (random networks)

Im Folgenden betrachten wir die Zufallsnetzwerke. Die einzelnen Knoten werden von den Kanten rein zufällig Muster aus ungerichteten Kanten, das ein Knoten

$p =$

Verteilung



$$P(k) = N(k)/N$$

## Einführung in die Theorie der komplexen Netzwerke



Ein Netzwerk wird formal mittels graphentheoretischer Größen beschrieben. Wir nehmen im Folgenden zunächst ein ungerichtetes, ungewichtetes Netzwerk bestehend aus einer Knotenart an, wobei  $N$  die Anzahl der Knoten und  $L$  die gesamte Anzahl der Links (Kanten) des Netzwerks ist. Jeder einzelne Knoten  $i$  ( $i \in \{1, 2, 3, \dots, N\}$ ) besitzt eine gewisse Anzahl von Verbindungskanten zu anderen Knoten und die Anzahl dieser Kanten bezeichnet man als den Knotengrad  $k_i$  des Knotens. Der durchschnittliche Knotengrad  $\langle k \rangle$  eines ungerichteten Netzwerks ist  $\langle k \rangle = \frac{2L}{N}$  und die maximale Anzahl möglicher Kanten ( $L_{max}$  (ein sogenannter vollständiger Graph) ist  $L_{max} = \frac{N(N-1)}{2}$ ). Eine der wichtigsten Netzwerk-charakterisierenden Größen ist die Verteilungsfunktion der Knotengrade  $P(k) = N(k)/N$ , wobei  $N(k)$  die Anzahl der Knoten mit Knotengrad  $k$  beschreibt. Weitere wichtige graphentheoretische Größen werden in dem Jupyter Notebook: Einführung in die Theorie der komplexen Netzwerke und Albert-Laszlo Barabasi, Network science, Chapter 2 Graph Theory.

# Einführung Teil II

Physik der sozio-ökonomischer Systeme mit dem Computer von Dr.phil.nat.Dr.rer.pol. Matthias Hanauske

[Home](#) [Research](#) [Contact](#)

[Einführung](#)

[Teil I](#)

[Teil II](#)

[Teil III](#)

[E-Learning](#)

## **II.1 Theorie der sozio-ökonomischen Netzwerke**

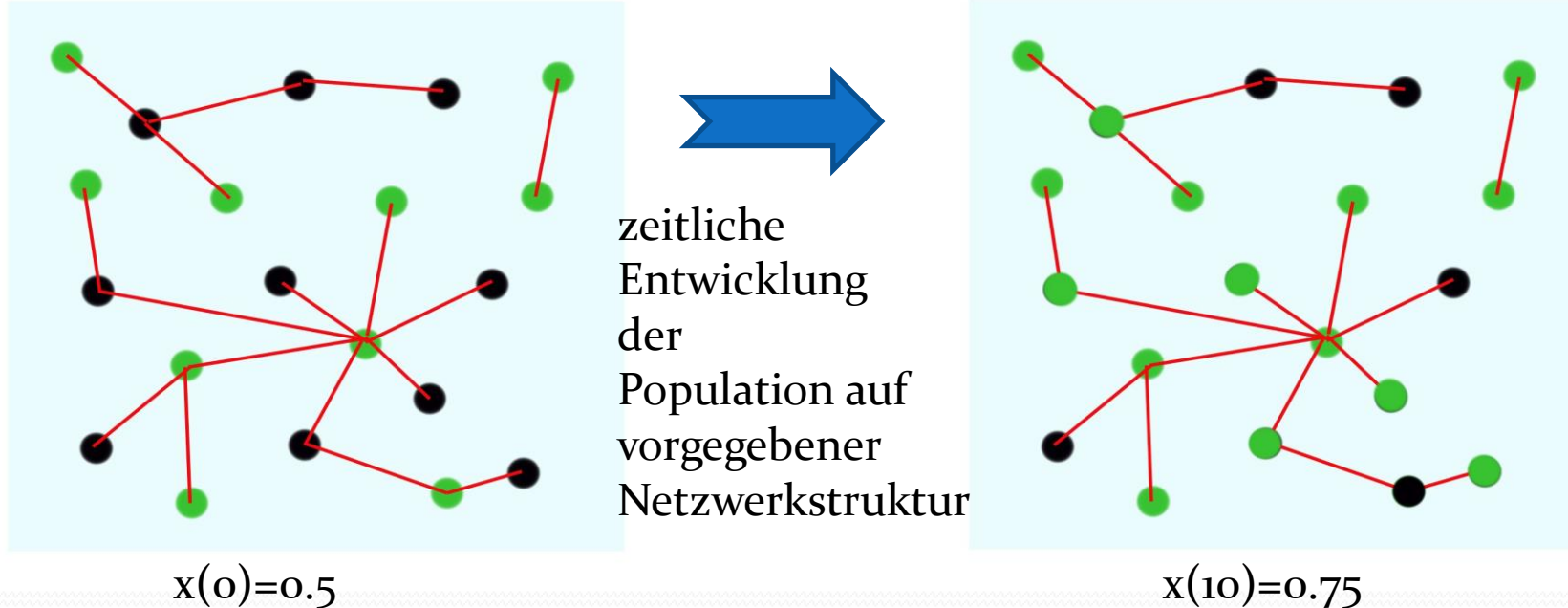
Eine bedeutende Einschränkung der deterministischen, evolutionären Spieltheorie ist deren zugrundeliegende Netzwerkstruktur (Netzwerktopologie). Die jeweiligen Spieler der betrachteten Population suchen in jeder Spielperiode einen neuen Spielpartner, wobei sie hierbei zufällig vorgehen (zufälliges Netzwerk) und vom Prinzip her mit jedem Spieler innerhalb der Population potentiell das zugrundeliegende Spiel spielen können (vollständig verbundenes Netzwerk). In Bimatrix Spielen suchen sich die Spieler der Teilpopulation A einen zufälligen Spielpartner aus Gruppe B (bzw. umgekehrt). Betrachtet man sich jedoch real existierende sozio-ökonomische Netzwerke, so zeigt sich, dass diese Annahme oft nicht erfüllt ist. Personen kennen oft nur eine Teilmenge von Spielern innerhalb der Population (kein vollständig verbundenes Netzwerk) und die Wahl der potentiellen Spielpartner erfolgt oft auch nicht nach zufälligen Mustern.

## **II.2 Einführung in die Theorie der komplexe Netzwerke**

Die Theorie der komplexen Netzwerke bildet die Grundlage zur Beschreibung einer Vielzahl von unterschiedlichen biologischen und sozio-ökonomischen Systemen. Die mathematische Beschreibung komplexer Netzwerke ist in mehreren Übersichtsartikel (siehe z.B. <http://barabasi.com/f/103.pdf>, <https://arxiv.org/abs/cond-mat/0106144>, <http://epubs.siam.org/doi/pdf/10.1137/S003614450342480>) zusammengefasst (siehe auch [Claudius Gros, "Complex and Adaptive Dynamical Systems, a Primer"](#)), wobei im

# Evolutionäre Spieltheorie auf komplexen Netzwerken

Viele in der Realität vorkommende evolutionäre Spiele werden auf einer definierten Netzwerkstruktur (Topologie) gespielt. Die Spieler der betrachteten Population sind hierbei nicht gleichwertig, sondern wählen als Spielpartner nur mit ihnen durch das Netzwerk verlinkte (verbundene) Partner aus.



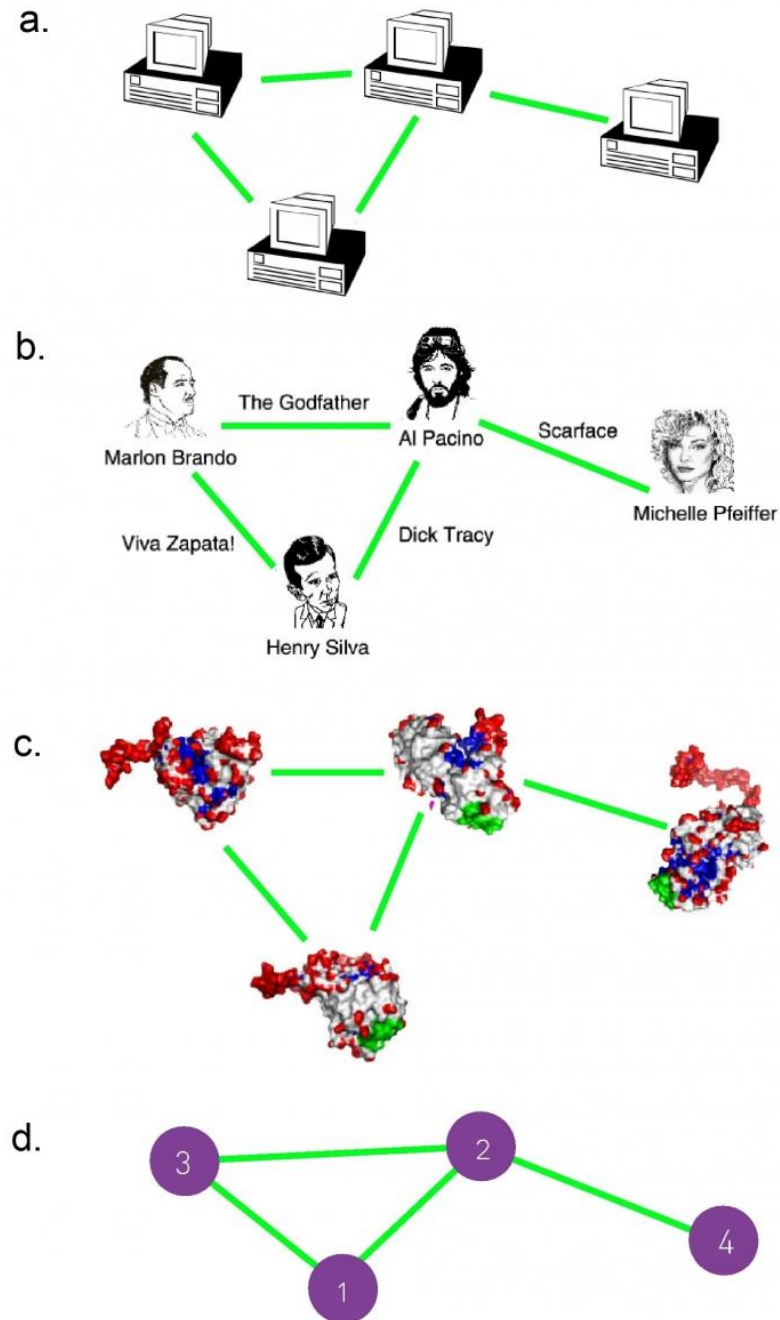
Mögliche Strategien: (grün, schwarz), Parameter  $t$  stellt die „Zeit“ dar.  
 $x(t)$  : Anteil der Spieler, die im Zeitpunkt  $t$  die Strategie „grün“ spielen.  
Die roten Verbindungslinien beschreiben die möglichen Spielpartner des Spielers

# Theorie der komplexen Netzwerke (I)

Da die Theorie der komplexen Netzwerke aus dem mathematischen Zweig der *Graphentheorie* entstanden ist benutzt sie nicht die „mathematischen Vokabeln“ der Spieltheorie. Man spricht z.B. nicht von Spielern, sondern von **Knoten** (bzw. Vertices). Die Verbindungen zwischen den Knoten werden als **Kanten** (bzw. Links) bezeichnet. Während die Spieler eines (klassischen) evolutionären Spiels mit allen anderen Spielern der Population in Kontakt treten können, ist dies bei einem Spiel auf einem komplexen Netzwerk im allgemeinen nicht möglich.



# Komplexe Netzwerke Knoten und Kanten



## Different Networks, Same Graph

The figure shows a small subset of (a) the Internet, where routers (specialized computers) are connected to each other; (b) the Hollywood actor network, where two actors are connected if they played in the same movie; (c) a protein-protein interaction network, where two proteins are connected if there is experimental evidence that they can bind to each other in the cell. While the nature of the nodes and the links differs, these networks have the same graph representation, consisting of  $N = 4$  nodes and  $L = 4$  links, shown in (d).

Abbildung entnommen von:

Network Science by Albert-László Barabási

<http://networksciencebook.com/>

Viele der im Teil 2 behandelten Themen sind in diesem Buch ausführlich behandelt.

# Netzwerke in der Realität

Netzwerke finden sich in den unterschiedlichsten sozialen, physikalischen und biologischen Systemen

- **Biologische Netzwerke**
  - Protein- und Gennetzwerke
- **Soziale Netzwerke**
  - Beziehungs- und Freundschaftsnetzwerke
  - Netzwerke von Geschäftsbeziehungen und Firmenbeteiligungen
  - Internetbasierte, soziale Web2.0 Netzwerke
- **Technologische Netzwerke**
  - Transportnetzwerke (Flug-, Zugrouten)
  - Internetverbindungen zwischen Computerservern
- **Informationsnetzwerke**
  - Wissensnetzwerke, Verlinkungen von Internetseiten
  - Zitationsnetzwerke von wissenschaftlichen Artikeln
  - Linguistische Netzwerke

# Theorie der komplexen Netzwerke (II)

Komplexe Netzwerke lassen sich wie folgt untergliedern:

- Handelt es sich nur um eine Knotenart (Spielergruppe), oder besteht das Netzwerk aus mehreren Knotenarten (z.B. Bi-Matrix Spiele).
- Sind die Kanten (Verbindungslinien zwischen den Knoten) gerichtet oder ungerichtet.
- Besitzen die Kanten zahlenmäßige Gewichtungen oder geben sie einfach an ob ein Knoten mit einem anderen verbunden oder nicht verbunden ist.
- Gibt es zeitliche Veränderungen des Netzwerks; ist die Anzahl der Knoten konstant oder wächst bzw. schrumpft sie im Laufe der Zeit.

# Theorie der komplexen Netzwerke (III)

(Beispiele unterschiedlicher komplexer Netzwerke)

- a) Nicht gerichtetes und ungewichtetes Netzwerk einer einzigen Knotenart.
- b) Nicht gerichtetes und ungewichtetes Netzwerk dreier verschiedener Knotenarten, wobei zusätzlich drei verschiedene Kantenarten existieren.
- c) Nicht gerichtetes aber gewichtetes Netzwerk. Sowohl die Knoten als auch die Kanten des Netzwerks besitzen zahlenmäßige Gewichtungen.
- d) Gerichtetes aber nicht gewichtetes Netzwerk. Es existiert nur eine Knoten- und gerichtete Kantenart.

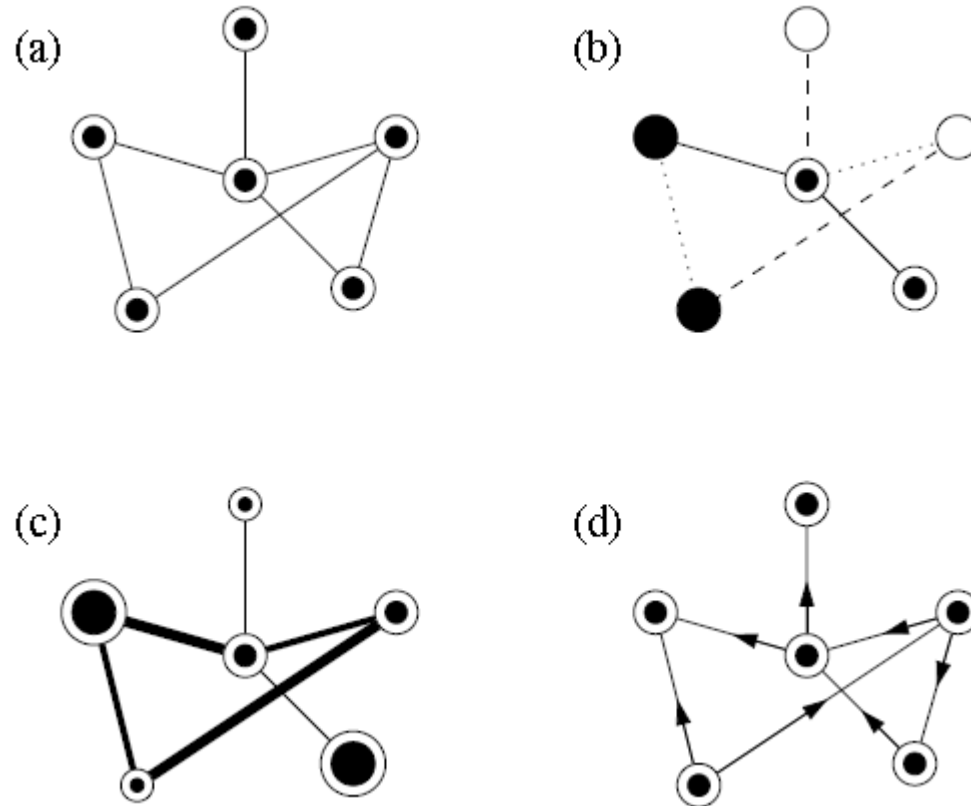


Abbildung: Unterschiedliche Netzwerktypen  
Die Abbildung ist dem folgenden Artikel entnommen:  
M. E. J. Newman,  
„The structure and function of complex networks”

# Theorie der komplexen Netzwerke (IV)

(Größen die ein Netzwerk charakterisieren)

- **Der Knotengrad  $k_i$**

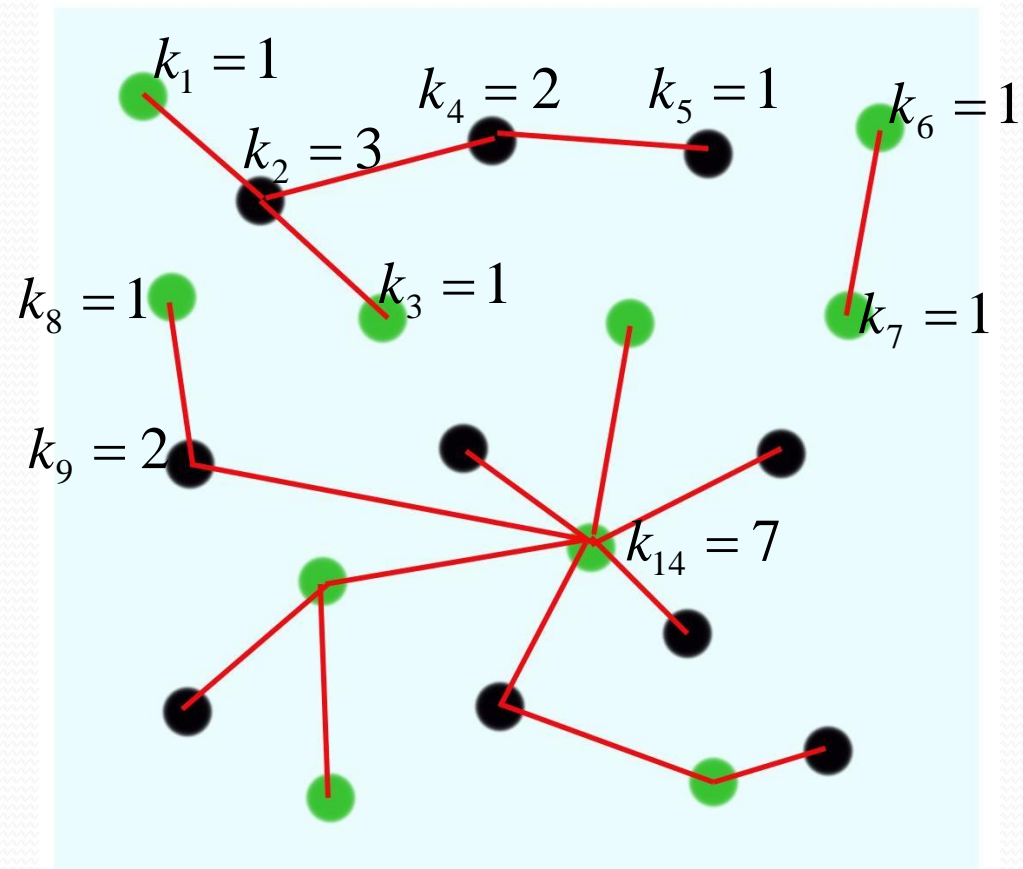
Der Knotengrad des Knotens  $i$  ist gleich der Anzahl der Kanten die der Knoten  $i$  besitzt. Bei gerichteten Netzwerken unterscheidet man zwischen dem eingehenden und ausgehenden Knotengrad. Bei gewichteten Netzwerken summiert man über die Zahlenfaktoren der gewichteten Kanten.

- **Der Clusterkoeffizient  $C_i$**

Der Clusterkoeffizient gibt die Wahrscheinlichkeit an, dass zwei nächste Nachbarn eines Knotens ebenfalls nächste Nachbarn untereinander sind. Der globale Wert  $C$  des Netzwerks stellt demnach eine Art von „Cliques“-Nachbarschafts-Eigenschaft des Netzwerks dar

- **Der Durchmesser des Netzwerks**

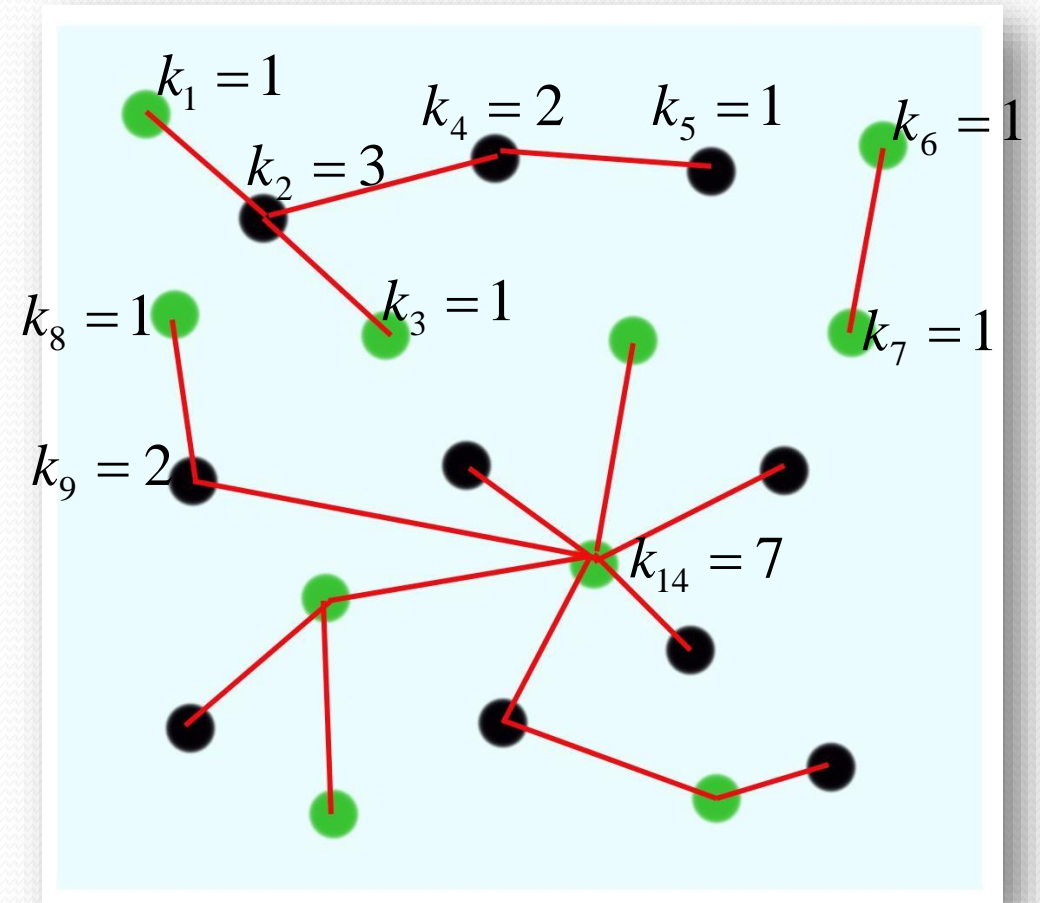
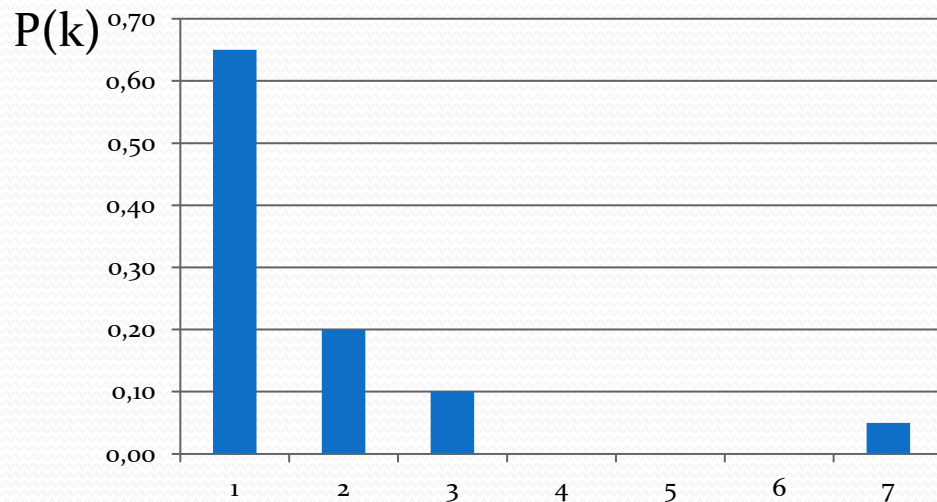
Der Durchmesser des Netzwerks gibt die maximale kürzeste Kantenlänge zwischen zwei beliebigen Knoten des Netzwerkes an.



# Theorie der komplexen Netzwerke (V)

## (Die Verteilungsfunktion der Knotengrade)

Die Verteilungsfunktion der Knotengrade  $P(k)$  (bzw.  $N(k)$ ) ist eine wichtige das Netzwerk charakterisierende Größe. Sie gibt an, wie groß der Anteil an Netzwerkknoten mit Knotengrad  $k$  ist. Bei realen (endlichen) Netzwerken ist diese Funktion keine kontinuierliche, sondern eine diskrete Funktion. In dem rechten Beispiel besitzt die Verteilungsfunktion das folgende Aussehen:



$k$

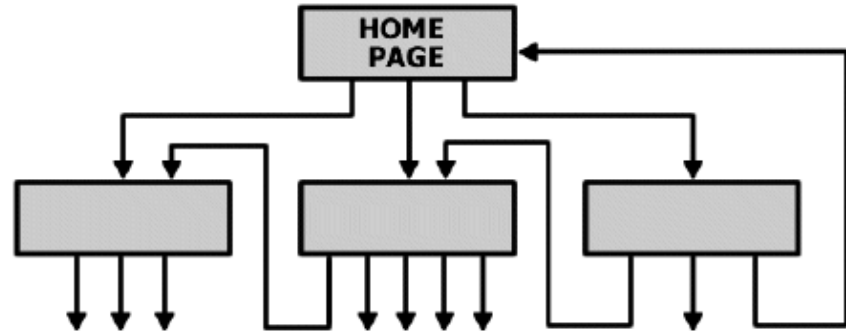
# The Structure and Function of Complex Networks\*

M. E. J. Newman<sup>†</sup>

## Netzwerkstrukturen in unterschiedlichsten Systemen

	Network	Type	$n$	$m$
Social	film actors	undirected	449 913	25 516 482
	company directors	undirected	7 673	55 392
	math coauthorship	undirected	253 339	496 489
	physics coauthorship	undirected	52 909	245 300
	biology coauthorship	undirected	1 520 251	11 803 064
	telephone call graph	undirected	47 000 000	80 000 000
	email messages	directed	59 912	86 300
	email address books	directed	16 881	57 029
	student relationships	undirected	573	477
	sexual contacts	undirected	2 810	
Information	WWW nd.edu	directed	269 504	1 497 135
	WWW Altavista	directed	203 549 046	2 130 000 000
	citation network	directed	783 339	6 716 198
	Roget's Thesaurus	directed	1 022	5 103
	word co-occurrence	undirected	460 902	17 000 000
Technological	Internet	undirected	10 697	31 992
	power grid	undirected	4 941	6 594
	train routes	undirected	587	19 603
	software packages	directed	1 439	1 723
	software classes	directed	1 377	2 213
	electronic circuits	undirected	24 097	53 248
	peer-to-peer network	undirected	880	1 296
Biological	metabolic network	undirected	765	3 686
	protein interactions	undirected	2 115	2 240
	marine food web	directed	135	598
	freshwater food web	directed	92	997
	neural network	directed	307	2 359

# WORLD-WIDE WEB



# INTERNET

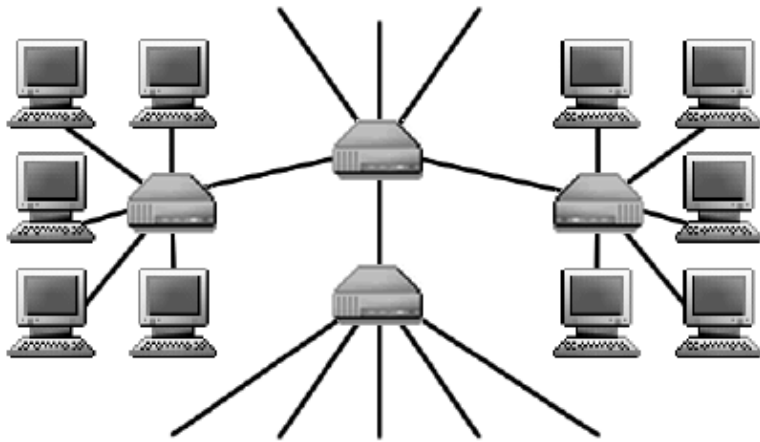
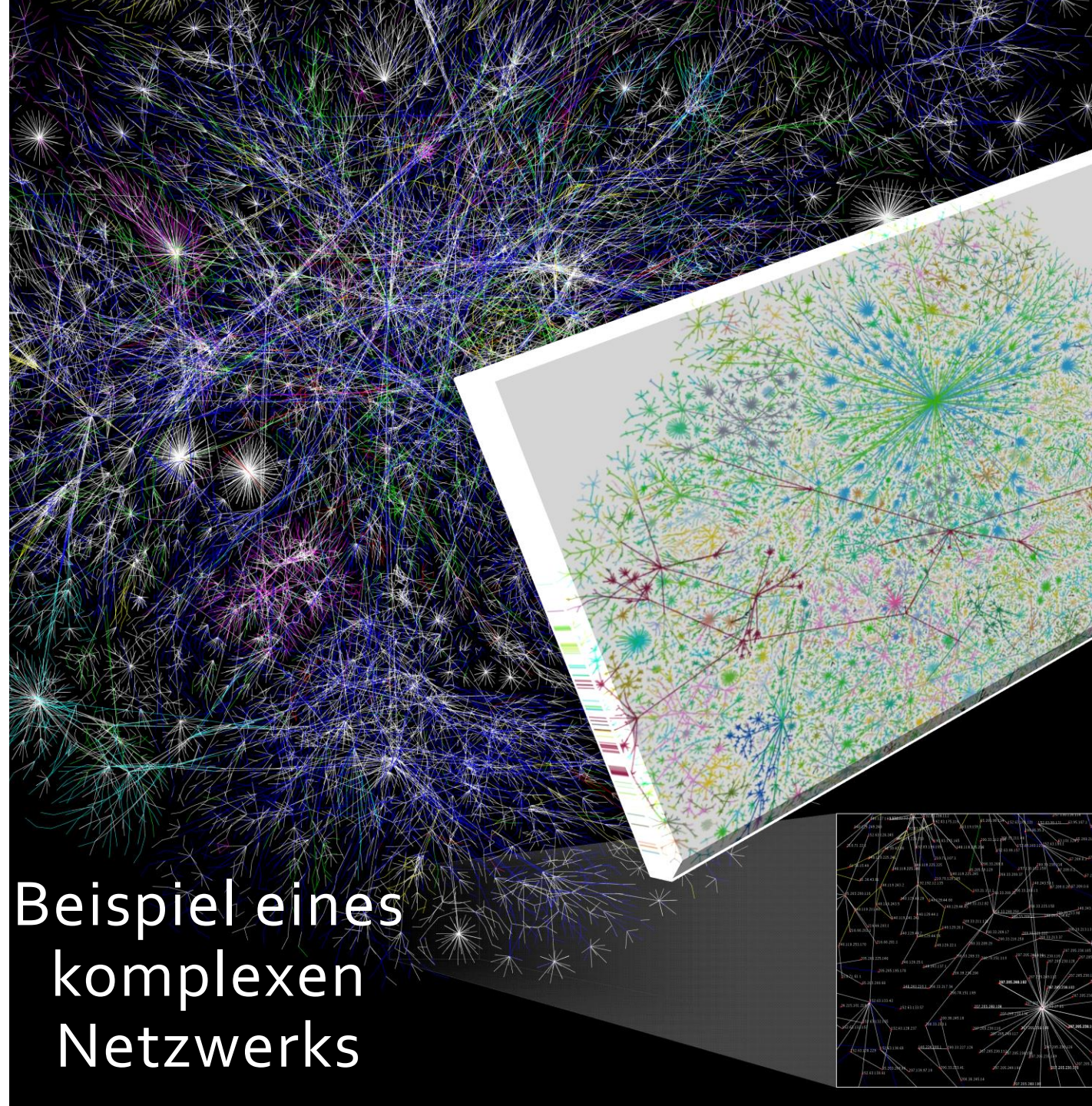


FIG. 1. Network structure of the World-Wide Web and the Internet. Upper panel: the nodes of the World-Wide Web are web documents, connected with directed hyperlinks (URLs). Lower panel: on the Internet the nodes are the routers and computers, the edges are the wires and cables that physically connect them. Figure courtesy of István Albert.

Beispiel eines  
komplexen  
Netzwerks

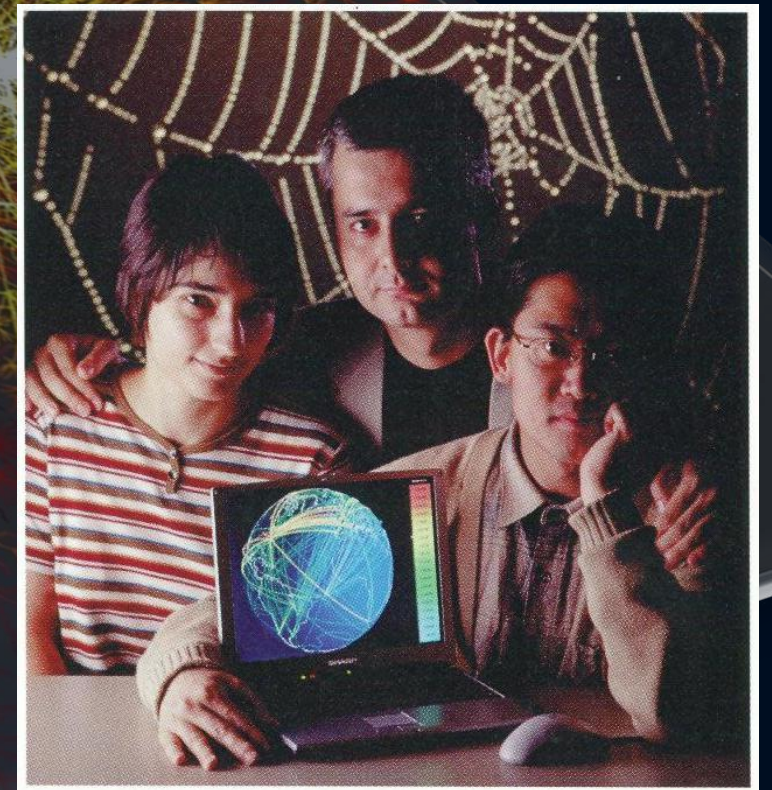




# Wechselwirkungen und Struktur im Internet

Im Jahre 1999 untersuchten Albert-Laszlo Barabasi und Mitarbeiter die topologische Struktur des Internets (WWW)

A photo taken for Business 2.0 magazine in 2000, showing Reka Albert, Hawoong Jeong and Albert-Laszlo Barabasi, soon after our publication of the paper on the topology of the WWW.  
(see <http://networksciencebook.com/>)



Viele der folgenden Abbildungen sind aus dem frei zugänglichen Buch „Network Science“ von Albert-Laszlo Barabasi entnommen.  
<http://networksciencebook.com/>

# Netzwerk-Klassen

Aufgrund ihrer unterschiedlichen Eigenschaften unterscheidet man die folgenden Netzwerk-Klassen:

**i. Zufällige Netzwerke**

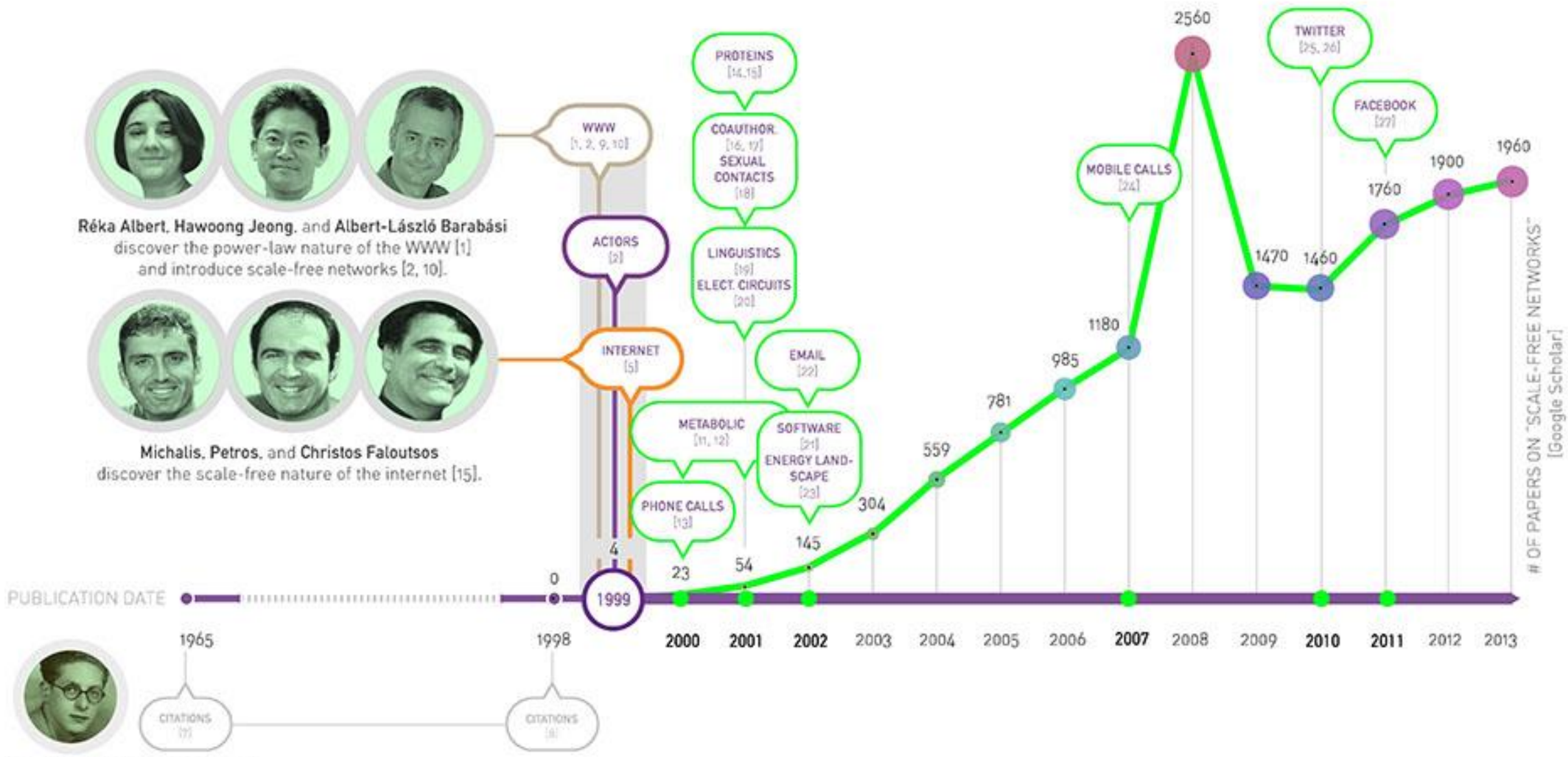
Die einzelnen Kanten bei *zufälligen Netzwerke* werden von den Knoten (Spielern) nach einem rein zufälligen Muster ausgewählt.

**ii. „Kleine Welt“-Netzwerke (small-world networks)**

- i. „Kleine Welt“-Netzwerke zeichnen sich durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten des Netzwerkes und einem großen Wert des Clusterkoeffizienten aus.

**iii. Exponentielle Netzwerke**

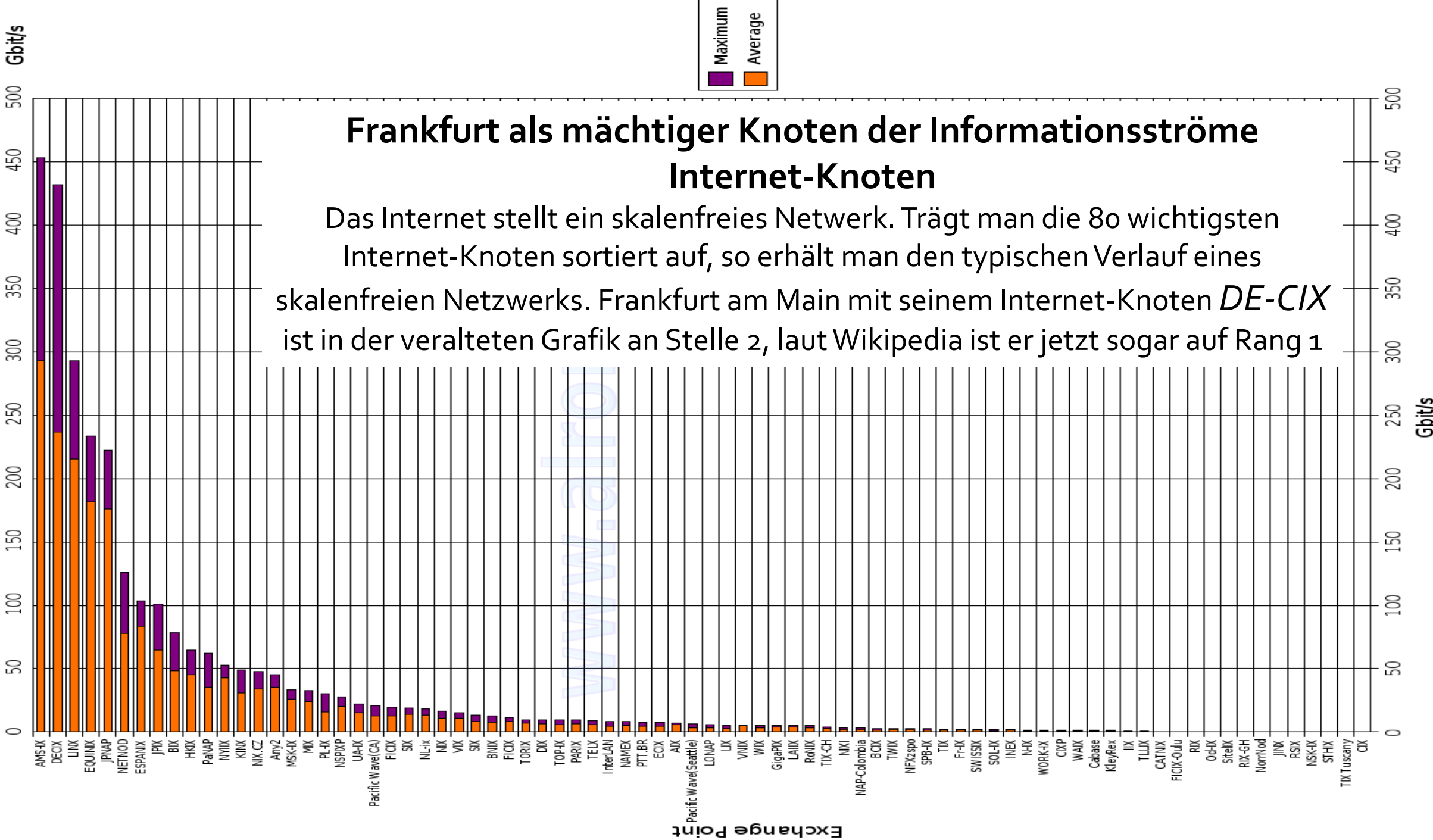
**iv. Skalenfreie Netzwerke**



**Derek de Solla Price [1922 - 1983]**  
 discovers that citations follow a power-law distribution [7], a finding later attributed to the scale-free nature of the citation network [2].

Skalenfreie Netzwerke auch in vielen weiteren sozio-ökonomischen und biologischen Systemen

Top 80 Internet exchange points

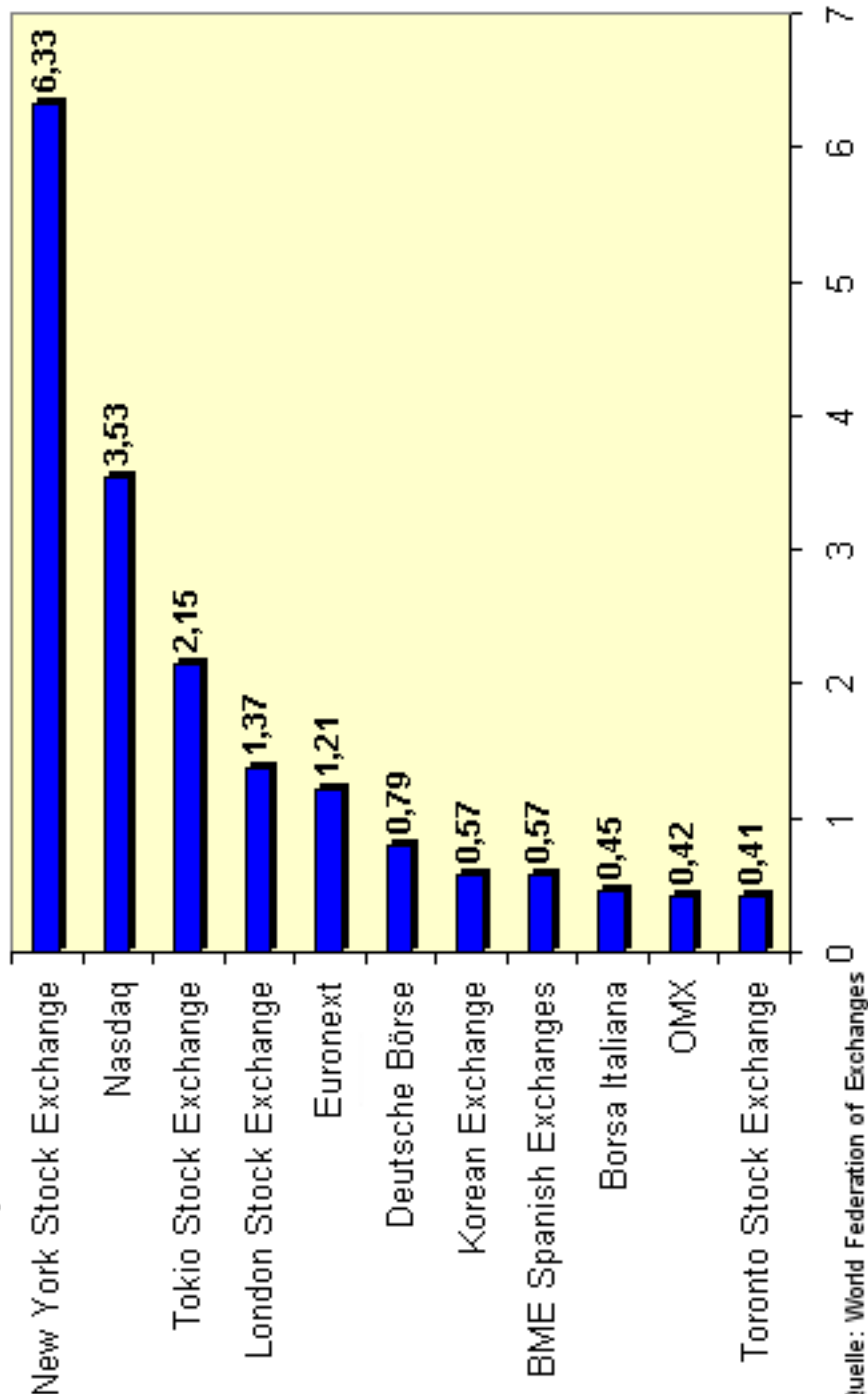


# Frankfurt als mächtiger Knoten der Informationsströme Internet-Knoten

Das Internet stellt ein skalenfreies Netzwerk. Trägt man die 80 wichtigsten Internet-Knoten sortiert auf, so erhält man den typischen Verlauf eines skalenfreien Netzwerkes. Frankfurt am Main mit seinem Internet-Knoten *DE-CIX* ist in der veralteten Grafik an Stelle 2, laut Wikipedia ist er jetzt sogar auf Rang 1

www.alto

**Die weltweit größten Wertpapierbörsen**  
nach Handelsvolumen (notierte inländische Aktien) in Billionen US-Dollar  
Januar bis April 2006



## Die Frankfurter Wertpapierbörse (FWB)

ist die bedeutendste deutsche Börse mit Sitz in Frankfurt am Main. Betreiberin und Träger ist die Deutsche Börse AG. Im Jahr 2000 wurde die Neue Börse im Industriehof in Frankfurt am Main in einem neuen Gebäude bezogen. Im Jahr 2005 wurden an den deutschen Börsen rund 3,8 Billionen Euro umgesetzt. Dabei entfielen vom Gesamtumsatz rund 3,2 Billionen Euro auf Aktien, Optionsscheine und börsengehandelte Fonds und rund 615 Milliarden Euro auf Anleihen. Der Aktienumsatz betrug 1,3 Billionen Euro, bei deutschen Aktien entfallen rund 98 % des Handels auf die Frankfurter Wertpapierbörse und Xetra, das elektronische Handelssystem der Deutschen Börse. Im Oktober 2008 entfielen 97 % der Umsätze in deutschen Aktien auf Xetra und die Frankfurter Parkettbörse. Bei ausländischen Aktien liefen über 86 % des Umsatzes über Xetra und den Präsenzhandel.'

## Contact

[Mailing list](#)

[Issue tracker](#)

[Source](#)

## Releases

### Stable (notes)

2.8.8 — November 2022

[download](#) | [doc](#) | [pdf](#)

### Latest (notes)

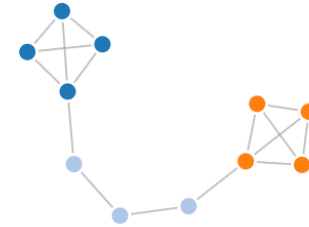
3.0 development

[github](#) | [doc](#) | [pdf](#)

### [Archive](#)



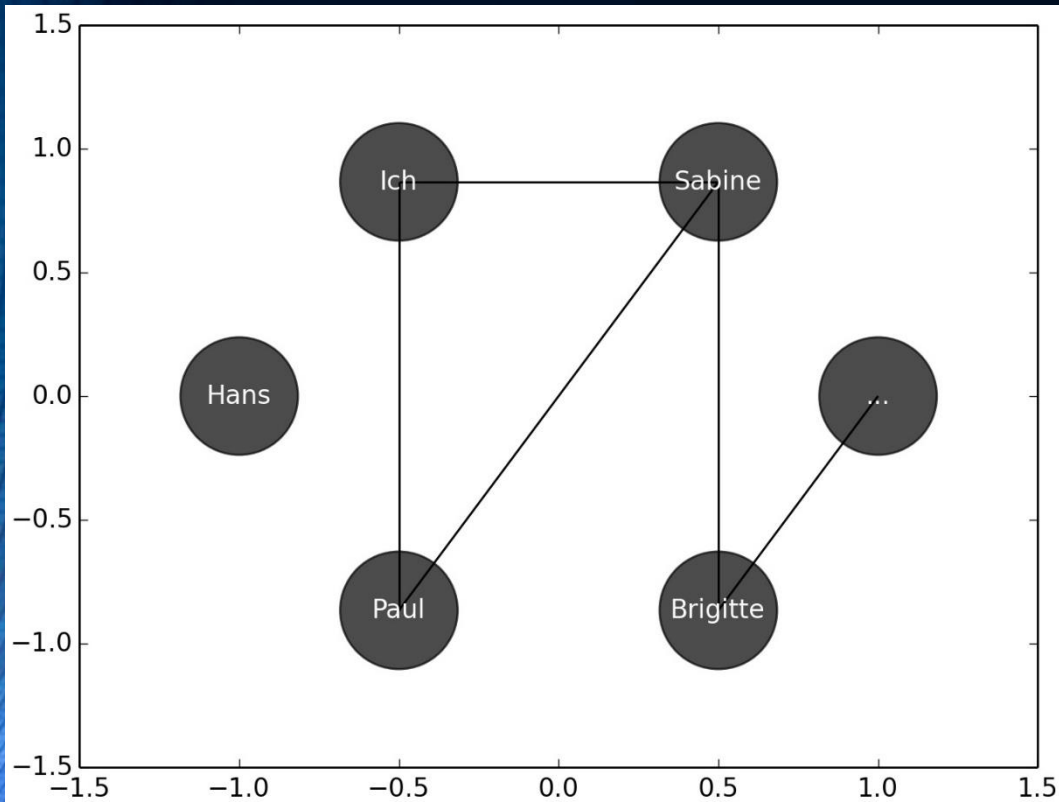
NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.



## Software for complex networks

- Data structures for graphs, digraphs, and multigraphs
- Many standard graph algorithms
- Network structure and analysis measures
- Generators for classic graphs, random graphs, and synthetic networks
- Nodes can be "anything" (e.g., text, images, XML records)
- Edges can hold arbitrary data (e.g., weights, time-series)
- Open source [3-clause BSD license](#)
- Well tested with over 90% code coverage
- Additional benefits from Python include fast prototyping, easy to teach, and multi-platform

# Simulation und Darstellung von komplexen Netzwerken mit Python (Teil II)



```
import networkx as nx
import matplotlib.pyplot as plt
```

```
G=nx.Graph()
```

```
#Hinzufuegen von Knoten zum Netzwerk
```

```
G.add_nodes_from(["Ich", "Paul", "Brigitte", "Sabine", "Hans", "..."])
```

```
G.add_edge("Ich", "Paul")
```

```
G.add_edge("Ich", "Sabine")
```

```
G.add_edge("Sabine", "Brigitte")
```

```
G.add_edge("Sabine", "Paul")
```

```
G.add_edge("Paul", "Sabine")
```

```
G.add_edge("Brigitte", "...")
```

```
#Erzeugung des Netzwerk-Bildes
```

```
graph_pos=nx.shell_layout(G)
```

```
#graph_pos=nx.shell_layout(G)
```

```
#graph_pos=nx.spring_layout(G)
```

```
nx.draw_networkx_nodes(G, graph_pos, node_size=3000, alpha=0.7, node_color="black")
```

```
nx.draw_networkx_edges(G, graph_pos)
```

```
nx.draw_networkx_labels(G, graph_pos, font_color="white")
```

```
#Speicherung des Bildes als .jpg
```

```
saveFig="./Netzwerk.jpg"
```

```
plt.savefig(saveFig, dpi=200, bbox_inches="tight", pad_inches=0.05, format="jpg")
```

```
plt.show()
```

# Physik der sozio-ökonomischen Systeme mit dem Computer

## (Physics of Socio-Economic Systems with the Computer)

Vorlesung gehalten an der J.W.Goethe-Universität in Frankfurt am Main

(Sommersemester 2024)

von Dr.phil.nat. Dr.rer.pol. Matthias Hanauske

Frankfurt am Main 01.02.2024

Zweiter Vorlesungsteil:

Einführung in die Theorie der komplexen Netzwerke

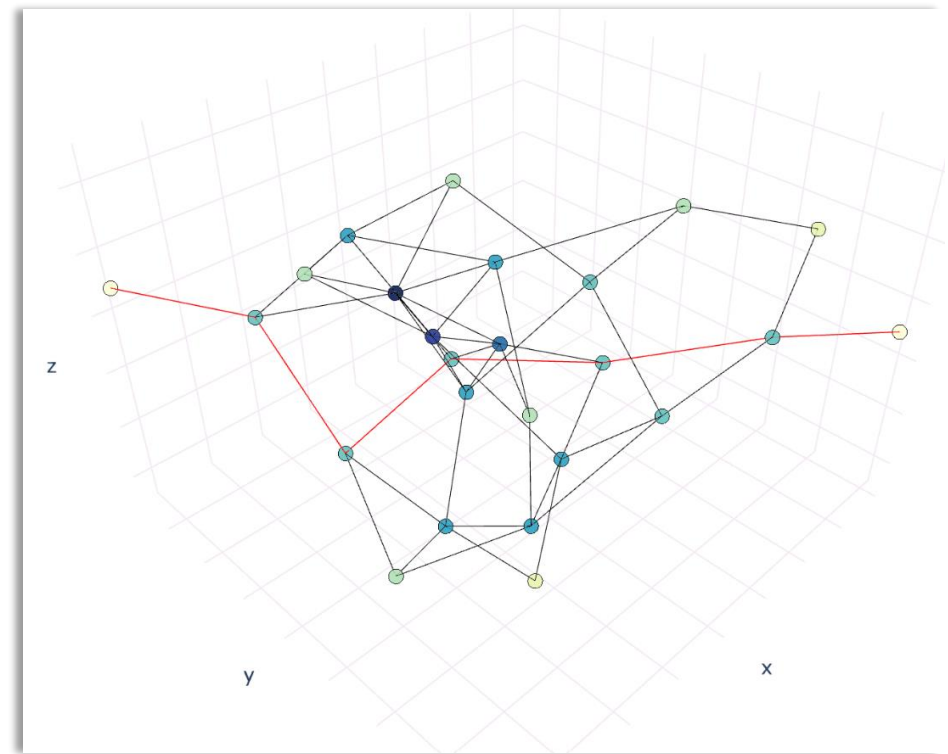
### Einführung

Eine bedeutende Einschränkung der deterministischen, evolutionären Spieltheorie ist deren zugrundeliegende Netzwerkstruktur (Netzwerktopologie). Die jeweiligen Spieler der betrachteten Population suchen in jeder Spielperiode einen neuen Spielpartner, wobei sie hierbei zufällig vorgehen (zufälliges Netzwerk) und vom Prinzip her mit jedem Spieler innerhalb der Population potenziell das zugrundeliegende Spiel spielen können (vollständig verbundenes Netzwerk). In Bimatrix Spielen suchen sich die Spieler der Teilpopulation A einen zufälligen Spielpartner aus Gruppe B (bzw. umgekehrt). Betrachtet man sich jedoch real existierende sozio-ökonomische Netzwerke, so zeigt sich, dass diese Annahme oft nicht erfüllt ist. Personen kennen oft nur eine Teilmenge von Spielern innerhalb der Population (kein vollständig verbundenes Netzwerk) und die Wahl der potenziellen Spielpartner erfolgt oft auch nicht nach zufälligen Mustern.

Die Theorie der komplexen Netzwerke bildet die Grundlage zur Beschreibung einer Vielzahl von unterschiedlichen biologischen und sozio-ökonomischen Systemen. Die formale/mathematische Beschreibung komplexer Netzwerke ist der Graphentheorie zuzuordnen. In dieser auf graphentheoretischen Grundlagen basierenden mathematischen Beschreibung der komplexen Netzwerke werden physikalische bzw. soziale Interaktionen durch Verbindungskanten zwischen den jeweiligen Netzwerkknoten beschrieben. Die theoretische Netzwerkforschung befasst sich mit der Entstehung und Beschreibung dieser Netzwerke. Bei einigen Modellnetzwerken können analytische Ergebnisse gewonnen werden. Die Anwendung der Theorie auf real existierende Netzwerkstrukturen ist ein sehr aktuelles interdisziplinäres Forschungsgebiet. Neben sozialen Netzwerken wie z.B. wissenschaftliche Kollaborationen, Koautorenschaften und Zitationsverflechtungen wissenschaftlicher Artikel, Kommunikationsnetzwerken wie dem Internet und diversen weiteren sozio-ökonomischen Netzwerkstrukturen werden mithilfe des mathematischen Modells der komplexen Netzwerke auch biologische Netzwerken wie z.B. neuronale oder Proteinnetzwerke beschrieben und analysiert.

Im Folgenden werden die Grundlagen der Theorie der komplexen Netzwerke an mehreren einfachen Beispielen beschrieben. Mittels Python und der Programmbibliothek [NetworkX](#) kann man in relativ einfacher Weise die unterschiedlichen Arten von komplexen Netzwerken grafisch darstellen und analysieren.

Da die Theorie der komplexen Netzwerke aus dem mathematischen Zweig der Graphentheorie entstanden ist, benutzt sie nicht die mathematischen Vokabeln der Spieltheorie. Man spricht z.B. nicht von Spielern





# Wichtige Netzwerk charakterisierende Größen

Teil II.2 Einführung in die Theorie der komplexe Netzwerke - Mozilla Firefox

Teil II.2 Einführung in die T x +

https://itp.uni-frankfurt.de/~hاناuske/VPSOC/T2/II-2.html

170%

☆ 📄 🌐 ☰

Meistbesucht Getting Started

- **Der Knotengrad  $k_i$**

Der Knotengrad des Knotens  $i$  ist gleich der Anzahl der Kanten die der Knoten  $i$  besitzt. Bei gerichteten Netzwerken unterscheidet man zwischen dem eingehenden  $k_i^{\text{in}}$  und ausgehenden Knotengrad  $k_i^{\text{out}}$ . Bei gewichteten Netzwerken summiert man über die Zahlenfaktoren der gewichteten Kanten.

- **Der durchschnittliche Knotengrad  $\langle k \rangle$**

Der durchschnittliche Knotengrad eines ungerichteten Netzwerks  $\langle k \rangle$  ist gegeben durch

$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^N k_i = \frac{2L}{N}, \text{ wobei } L = \text{die gesamte Anzahl der Links (Kanten) des Netzwerks ist.}$$

- **Die Nachbarschaftsmatrix (Adjazenzmatrix)  $A_{ij}$**

Die Nachbarschaftsmatrix des Netzwerkes beschreibt, welche Knoten des Graphen durch eine Kante verbunden sind. Zeigt eine Kante von Knoten  $j$  auf den Knoten  $i$ , dann ist der zugehörige Matrixwert  $A_{ij} = 1$ , wohingegen ein Eintrag  $A_{ij} = 0$  bedeutet, dass keine Kante von  $j$  nach  $i$  zeigt. Bei ungerichteten Netzwerken ist die Nachbarschaftsmatrix symmetrisch. Der Knotengrad  $k_i$  des Knotens  $i$

(bzw.  $k_i^{\text{in}}$  und  $k_i^{\text{out}}$  bei gerichteten Graphen) lässt sich mittels der Nachbarschaftsmatrix wie folgt bestimmen:  $k_i = \sum_{j=1}^N A_{ij}$  ( $k_i^{\text{in}} = \sum_{j=1}^N A_{ij}$  und  $k_i^{\text{out}} = \sum_{j=1}^N A_{ji}$  bei gerichteten Graphen), wobei

die gesamte Anzahl der Kanten sich wie folgt bestimmen lässt:  $L = \sum_{i,j=1}^N A_{ij}/2$  und die Größe  $N$  die Anzahl der Knoten des Netzwerkes bezeichnet. In gewichteten Netzwerken sind die Einträge der Nachbarschaftsmatrix nicht mehr auf Null und Eins beschränkt  $A_{ij} = w_{ij}$ .

- **Die Verteilungsfunktion der Knotengrade  $P(k) = N(k)/N$**

Nachbarschaftsmatrix nicht mehr auf Null und Eins beschränkt  $A_{ij} = w_{ij}$ .

- **Die Verteilungsfunktion der Knotengrade**  $P(k) = N(k)/N$

Die Verteilungsfunktion der Knotengrade  $P(k)$  (bzw.  $N(k)$ ) ist eine wichtige das Netzwerk charakterisierende Größe. Sie gibt an, wie groß der Anteil an Netzwerkknoten mit Knotengrad  $k$  ist. Bei realen (endlichen) Netzwerken ist diese Funktion keine kontinuierliche, sondern eine diskrete Funktion  $P(k) \approx P_k$ . Bezeichnen wir mit  $N_k$  die Anzahl der Knoten mit Knotengrad  $k$ , so gilt  $P_k = \frac{N_k}{N}$ . Es gilt  $\sum_{k=0}^{\infty} P_k = 1$  und der durchschnittliche Knotengrad berechnet sich wie folgt  $\langle k \rangle = \sum_{k=0}^{\infty} k P_k$ .

- **Die maximale Anzahl möglicher Kanten**  $L_{max}$

Die maximale Anzahl möglicher Kanten in einem ungerichteten Netzwerk (ein sogenannter 'complete graph') besitzt  $L_{max} = \frac{N(N-1)}{2}$  Kanten. Viele real existierende Netzwerke sind dünnbesetzt ( $L \ll L_{max}$ ).

- **Die kürzeste Verbindungstrecke zwischen zwei Knoten**  $d_{ij}$

Die Anzahl der Kanten die in einem Verbindungsweg von Knoten  $i$  zum Knoten  $j$  durchlaufen wird hängt vom gewählten Pfad ab. Die kürzeste Verbindungstrecke  $d_{ij}$  kann hierbei im allgemeinen auf unterschiedlichen Wegen realisiert sein, wobei der Pfad keine Schleifen enthalten darf und sich nicht kreuzen darf. Die Anzahl der kürzeste Verbindungswege  $N_{ij}$  lässt sich mittels der Nachbarschaftsmatrix berechnen. Existieren z.B. Verbindungswege vom Knoten  $i$  zum Knoten  $j$  mit  $d_{ij} = 2$ , so muss die Anzahl der kürzeste Verbindungswege  $N_{ij} = \sum_{l=1}^N A_{il} A_{jl} \geq 1$  sein.

- **Der Clusterkoeffizient**  $C_i$  bzw.  $C$

Der Clusterkoeffizient gibt die Wahrscheinlichkeit an, dass zwei nächste Nachbarn eines Knotens

berechnen. Existieren z.B. Verbindungswege vom Knoten  $i$  zum Knoten  $j$  mit  $w_{ij} = 2$ , so muss die

Anzahl der kürzeste Verbindungswege  $N_{ij} = \sum_{l=1}^N A_{il}A_{jl} \geq 1$  sein.

- **Der Clusterkoeffizient  $C_i$  bzw.  $C$**

Der Clusterkoeffizient gibt die Wahrscheinlichkeit an, dass zwei nächste Nachbarn eines Knotens ebenfalls nächste Nachbarn untereinander sind. Für einen speziellen Knoten  $i$  berechnet er sich mittels:

$C_i = \frac{2L_i}{k_i(k_i-1)}$ , wobei  $L_i$  die Anzahl der Kanten darstellt, die die nächsten Nachbarn des Knoten  $i$

miteinander verbinden. Der globale Wert  $C$  des Clusterkoeffizienten ist der Mittelwert aller  $C_i$ 's ( $C = \frac{1}{N} \sum_{i=1}^N C_i$ ) und stellt demnach eine Art von der Enge der Nachbarschaftsbeziehungen des

Netzwerks dar.

- **Durchschnittliche Anzahl der  $m$ -nächsten Nachbarn  $z_m$**

$z_1$  stellt hierbei den Wert der mittleren Knotenzahl des Netzwerkes dar ( $z_1 = \langle k \rangle$ ) und  $z_2$  die mittlere Anzahl zweiter-nächster Nachbarn ( $z_2 = \langle k^2 \rangle - \langle k \rangle$ ), siehe z.B. [Claudius Gros, "Complex and Adaptive Dynamical Systems, a Primer", S:18](#)).

- **Durchmesser des Netzwerks  $l$**

Der Durchmesser des Netzwerks gibt die maximale kürzeste Kantenlänge zwischen zwei beliebigen Knoten des Netzwerkes an:  $l = \log(N/z_1)/\log(z_2/z_1) + 1$ , wobei  $N$  die Anzahl der Knoten des Netzwerkes darstellt (siehe z.B. [Claudius Gros, "Complex and Adaptive Dynamical Systems, a Primer", S:20](#)). Der Wert  $l$  wird auch in einigen Lehrbüchern als  $d_{\max}$  bezeichnet (siehe [Albert-Laszlo Barabasi, Network science, Chapter 2 Graph Theory](#)).

- **Größter verbundener Knotenclusters  $N_G$  (Giant component, Hub)**

Die Anzahl der Knoten im größten verbundener Knotencluster des Netzwerkes ist mit  $N$

miteinander verbinden. Der globale Wert  $C$  des Clusterkoeffizienten ist der Mittelwert aller  $C_i$ 's ( $C = \frac{1}{N} \sum_{i=1}^N C_i$ ) und stellt demnach eine Art von der Enge der Nachbarschaftsbeziehungen des Netzwerks dar.

- **Durchschnittliche Anzahl der  $m$ -nächsten Nachbarn  $z_m$**

$z_1$  stellt hierbei den Wert der mittleren Knotenzahl des Netzwerkes dar ( $z_1 = \langle k \rangle$ ) und  $z_2$  die mittlere Anzahl zweiter-nächster Nachbarn ( $z_2 = \langle k^2 \rangle - \langle k \rangle$ , siehe z.B. [Claudius Gros, "Complex and Adaptive Dynamical Systems, a Primer", S:18](#)).

- **Durchmesser des Netzwerks  $l$**

Der Durchmesser des Netzwerks gibt die maximale kürzeste Kantenlänge zwischen zwei beliebigen Knoten des Netzwerkes an:  $l = \log(N/z_1)/\log(z_2/z_1) + 1$ , wobei  $N$  die Anzahl der Knoten des Netzwerkes darstellt (siehe z.B. [Claudius Gros, "Complex and Adaptive Dynamical Systems, a Primer", S:20](#)). Der Wert  $l$  wird auch in einigen Lehrbüchern als  $d_{\max}$  bezeichnet (siehe [Albert-Laszlo Barabasi, Network science, Chapter 2 Graph Theory](#)).

- **Größter verbundener Knotenclusters  $N_G$  (Giant component, Hub)**

Die Anzahl der Knoten im größten verbundenen Knotenclusters des Netzwerkes wird mit  $N_G$  bezeichnet.

Diese und weitere graphentheoretischen Größen (Number of Shortest Paths Between Two Nodes, Shortest Path, Average Path Length, ...) sind in [Albert-Laszlo Barabasi, Network science, Chapter 2 Graph Theory](#) zu finden.



# Zufällige Netzwerke

Verteilungsfunktion  
der Knotengrade

$$P(k)$$

bzw.  $N(k) := N * P(k)$

## VI. CLASSICAL RANDOM GRAPHS, THE ERDÖS-RÉNYI MODEL

The simplest and most studied network with undirected edges was introduced by Erdős and Rényi (ER model) [77,78]. In this network:

- (i) the total number of vertices,  $N$ , is fixed;
- (ii) the probability that two arbitrary vertices are connected equals  $p$ .

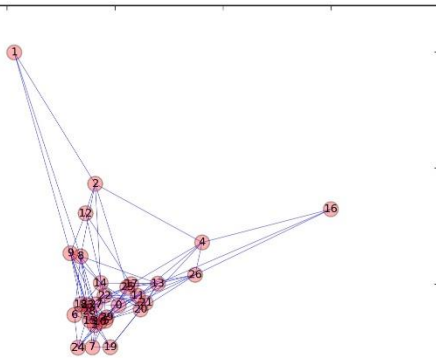
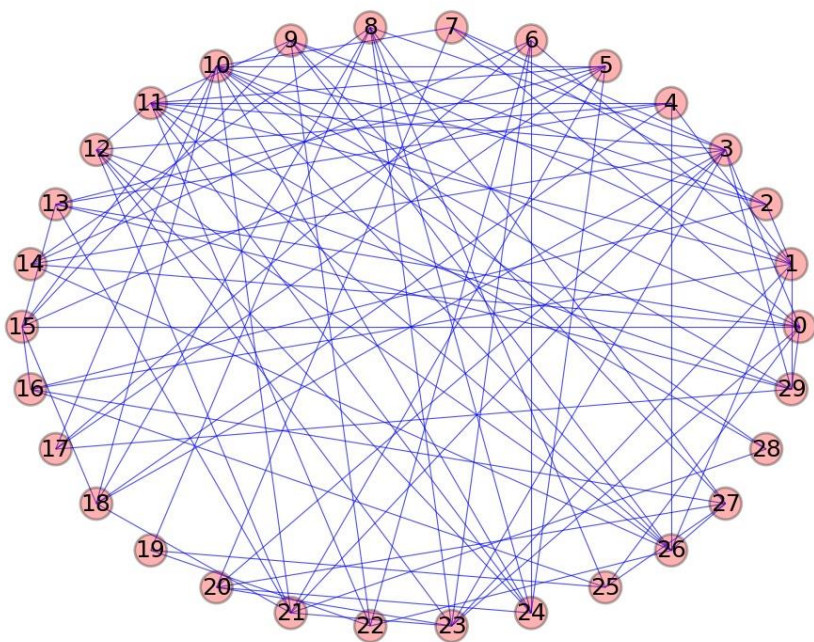
One sees that, on average, the network contains  $pN(N-1)/2$  edges. The degree distribution is binomial,

$$P(k) = \binom{N-1}{k} p^k (1-p)^{N-1-k}, \quad (4)$$

so the average degree is  $\bar{k} = p(N-1)$ . For large  $N$ , the distribution, Eq. (4) takes the Poisson form,

$$P(k) = e^{-\bar{k}} \bar{k}^k / k!. \quad (5)$$

# Python (Version 2)



```
import networkx as nx
import matplotlib.pyplot as plt
from random import randint
from random import uniform

G=nx.Graph()

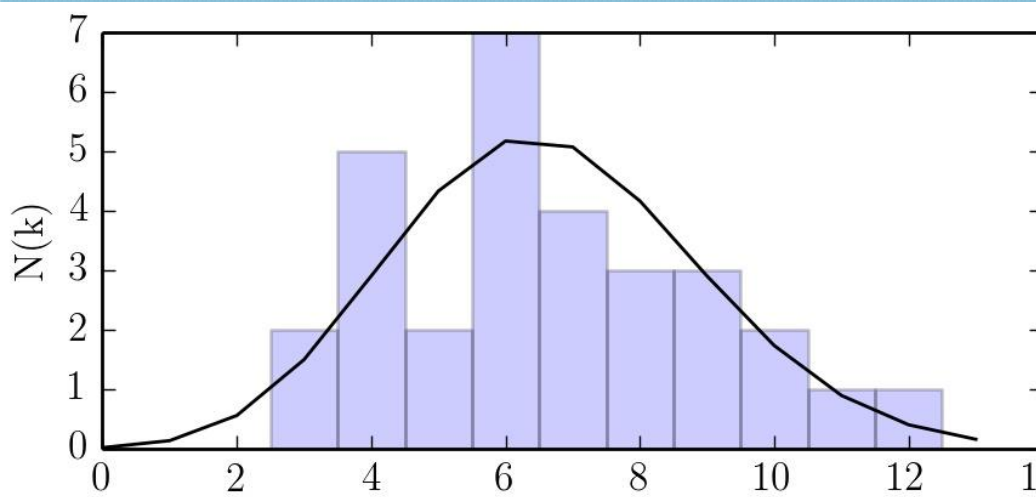
#Allgemeine Festlegungen des Netzwerks
NKn =30 #Anzahl der Knoten (vertices)
anzedges=100 #Gesamte Anzahl der im Netzwerk bestehenden Kanten (links, connections)

#Hinzufuegen der Knoten zum Netzwerk
G.add_nodes_from(range(0,NKn,1))

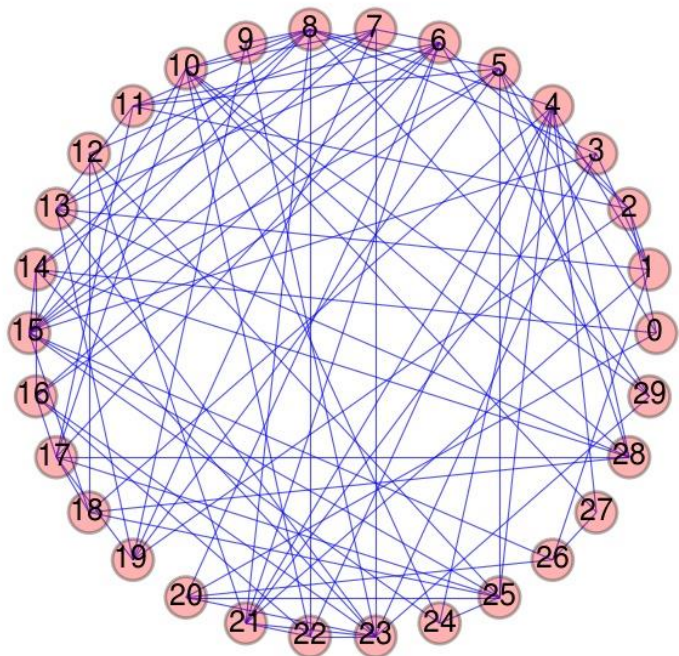
#Erzeugung der Kanten des Netzwerkes (zufaellig anzedges-Kanten zwischen den NKn-Knoten erzeugen)
links=0
while links < anzedges:
    KnA = randint(0, NKn-1)
    KnB = randint(0, NKn-1)
    if KnA != KnB and list(G.edges()).count((KnA,KnB))==0 and list(G.edges()).count((KnB,KnA))==0:
        G.add_edge(KnA,KnB)
        links=links+1

#Erzeugung des Netzwerk-Bildes
node_size=150|
node_alpha=0.3
node_color="red"
edge_tickness=0.4
edge_alpha=0.7
edge_color="blue"
node_text_size=9
text_font="sans-serif"
graph_pos=nx.shell_layout(G)
#graph_pos=nx.spectral_layout(G)
#graph_pos=nx.spring_layout(G)
nx.draw_networkx_nodes(G,graph_pos,node_size=node_size,alpha=node_alpha,node_color=node_color)
nx.draw_networkx_edges(G,graph_pos,width=edge_tickness,alpha=edge_alpha,edge_color=edge_color)
nx.draw_networkx_labels(G,graph_pos,font_size=node_text_size,font_family=text_font)

#Speicherung des Bildes als .jpg und .pdf Datei
saveFig="./Netzwerk.jpg"
```



$$N(k) = N \binom{N-1}{k} p^k (1-p)^{N-1-k}$$



$$N = 30, p = 0.2299, m = 100$$

```

import networkx as nx
import matplotlib.pyplot as plt
from random import randint
from random import uniform
import numpy as np
import matplotlib
import matplotlib.gridspec as gridspec
from matplotlib.ticker import NullFormatter

nullfmt = NullFormatter() # Keine Labels im Netzwerkplot

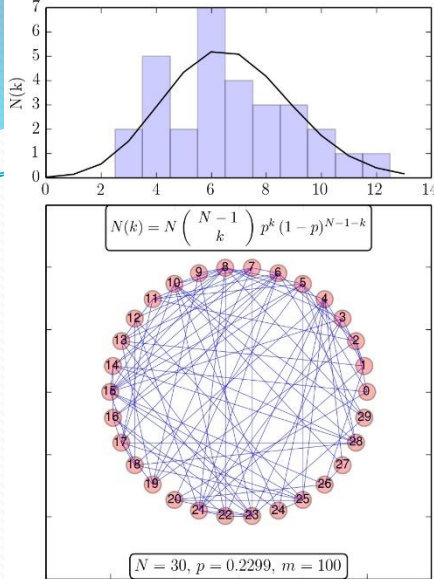
#Von https://de.wikibooks.org/wiki/Algorithmensammlung:_Statistik:_Binomialkoeffizient
def binomialkoeffizient(n, k):
    if k == 0: return 1
    if 2*k > n:
        ergebnis = binomialkoeffizient(n, n-k)
    else:
        ergebnis = n-k+1
        for i in range(2, k+1): # i in [2; k]
            ergebnis *= (n-k+i) # Selbstmultiplikation
            ergebnis /= i # Achtung: Ergebnis ist eine Kommazahl!
    return int(ergebnis)]

#Analytische Verteilungsfunktion eines zufaelligen Netzwerkes
def P(n, kmin, kmax, p):
    ergebnis=[]
    for k in range(kmin, kmax, 1):
        ergebnis.append(binomialkoeffizient(n-1, k)*p**k*(1-p)**(n-1-k))
    return ergebnis

#plot settings
params = {
    'figure.figsize' : [5, 7.2],
    'text.usetex' : True,
}
matplotlib.rcParams.update(params)

#Grid
plt.figure(0)
gs = gridspec.GridSpec(2, 1, height_ratios=[1,2.2], hspace=0.1)
ax1 = plt.subplot(gs[0])
ax2 = plt.subplot(gs[1])

```



```
#####  
#Beginn des eigentlichen Python Programms "Zufaelliches Netzwerk"  
G=nx.Graph()  
#Allgemeine Festlegungen des Netzwerks  
NKn =30 #Anzahl der Knoten (vertices)  
anzedges=100 #Gesamte Anzahl der im Netzwerk bestehenden Kanten (links, connections)  
  
#Hinzufuegen der Knoten zum Netzwerk  
G.add_nodes_from(range(0,NKn,1))  
  
#Erzeugung der Kanten des Netzwerkes (zufaellich anzedges-Kanten zwischen den NKn-Knoten erzeugen)  
links=0  
while links < anzedges:  
    KnA = randint(0, NKn-1)  
    KnB = randint(0, NKn-1)  
    if KnA != KnB and list(G.edges()).count((KnA,KnB))==0 and list(G.edges()).count((KnB,KnA))==0:  
        G.add_edge(KnA,KnB)  
        links=links+1  
  
#Liste der Knotengrade  
degree_sequence=sorted([d for n,d in G.degree()],reverse=True) # degree sequence  
maxk=np.max(degree_sequence)  
  
#Berechnung der Wahrscheinlichkeit das Knoten KnA mit KnB verbunden ist  
p=(2*anzedges/float(NKn*(NKn-1)))  
  
#Erzeugung des Bildes der Verteilungsfunktion der Knotengrade N(k) (analytisch,simulativ)  
ax1.plot(range(0,int(maxk+2),1),np.array(P(NKn,0,int(maxk+2),p))*NKn, linewidth=1, linestyle='-', c="black")  
ax1.hist(degree_sequence,bins=range(0,int(maxk+2),1), align="left", histtype='bar', color="blue", alpha=0.2)  
  
#Achsenbeschriftung  
ax1.set_ylabel(r'$\rm N(k)$')  
ax2.yaxis.set_major_formatter(nullfmt)  
ax2.xaxis.set_major_formatter(nullfmt)
```

```
#Erzeugung des Netzwerk-Bildes
```

```
node_size=150  
node_alpha=0.3  
node_color="red"  
edge_tickness=0.4  
edge_alpha=0.7  
edge_color="blue"  
node_text_size=9  
text_font="sans-serif"  
graph_pos=nx.shell_layout(G)  
nx.draw_networkx_nodes(G,graph_pos,node_size=node_size,alpha=node_alpha, node_color=node_color)  
nx.draw_networkx_edges(G,graph_pos,width=edge_tickness, alpha=edge_alpha,edge_color=edge_color)  
nx.draw_networkx_labels(G, graph_pos,font_size=node_text_size,font_family=text_font)
```

```
# Plotten der Netzwerkeigenschaften in das Bild
```

```
roundp="%.4f"%p  
textstr1=r'$N(k) = N \left( \begin{array}{c} N-1 \\ k \end{array} \right) p^k (1-p)^{N-1-k}$'  
textstr2=r'$N='+str(NKn)+' , p='+roundp+' , m='+str(anzedges)+'$'  
props = dict(boxstyle='round', facecolor='white', alpha=0.92)  
plt.text(0, 1.46, textstr1, fontsize=11, verticalalignment='top', horizontalalignment='center', bbox=props)  
plt.text(0, -1.46, textstr2, fontsize=12, verticalalignment='bottom', horizontalalignment='center', bbox=props)
```

```
#Speicherung des Bildes als .jpg und .pdf Datei
```

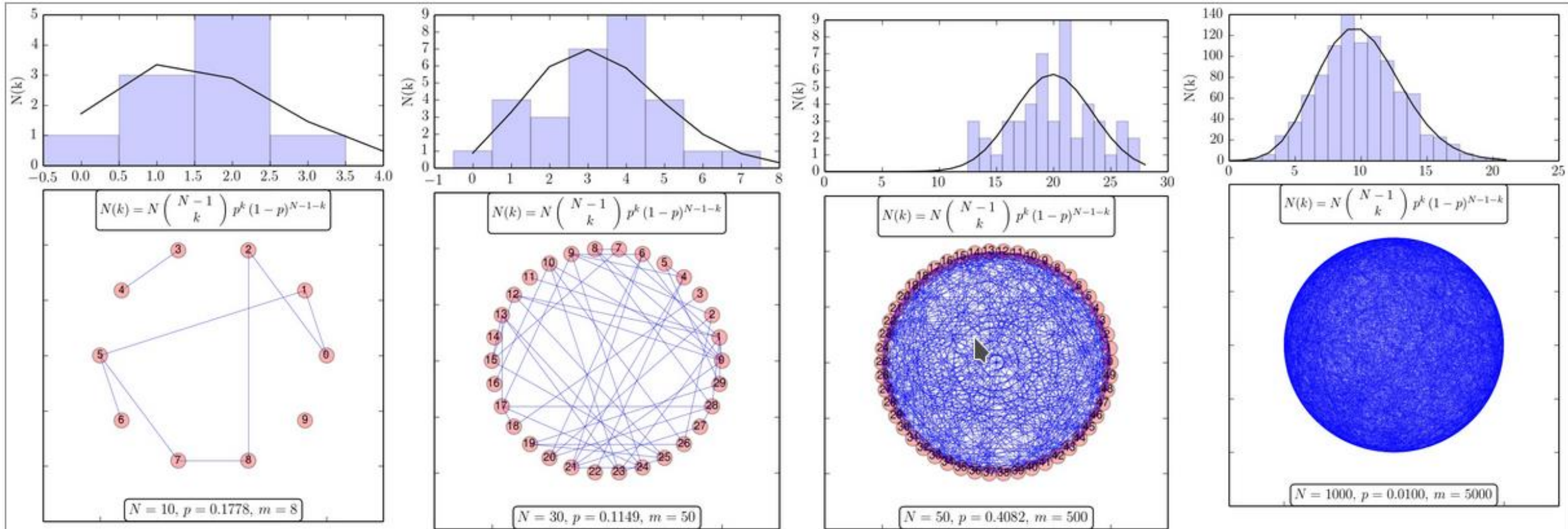
```
saveFig="./Netzwerk.jpg"  
plt.savefig(saveFig, dpi=200,bbox_inches="tight",pad_inches=0.05,format="jpg")  
saveFig="./Netzwerk.pdf"  
plt.savefig(saveFig,bbox_inches="tight",pad_inches=0.05,format="pdf")  
plt.show()
```

# Python (Version 3)



# Python Programm *RandomNetwork.py*

## II.2.1 Zufällige Netzwerke (Random Networks)



Ergebnisse des Python Skriptes RandomNetwork.py: Zufallsgraph und die zugehörige Verteilungsfunktion der Knotengrade (schwarze Kurve: analytische Verteilung).

# Python Programm VPSOC-RandomNetwork\_evol.py

Die einzelnen Kanten bei zufälligen Netzwerke werden von den Knoten (Spielern) nach einem rein zufälligen Muster ausgewählt. Im Erdos-Renyi Modell (Erdos and Renyi, 1959) werden  $N$  Knoten zufällig mit  $m$  ungerichteten Kanten verbunden. Die Wahrscheinlichkeit  $p$ , das ein Knoten mit dem anderen verbunden ist demnach  $p = 2m/(N(N - 1))$ . Die Verteilungsfunktion der Knotengrade  $P(k)$  ist binomialverteilt:

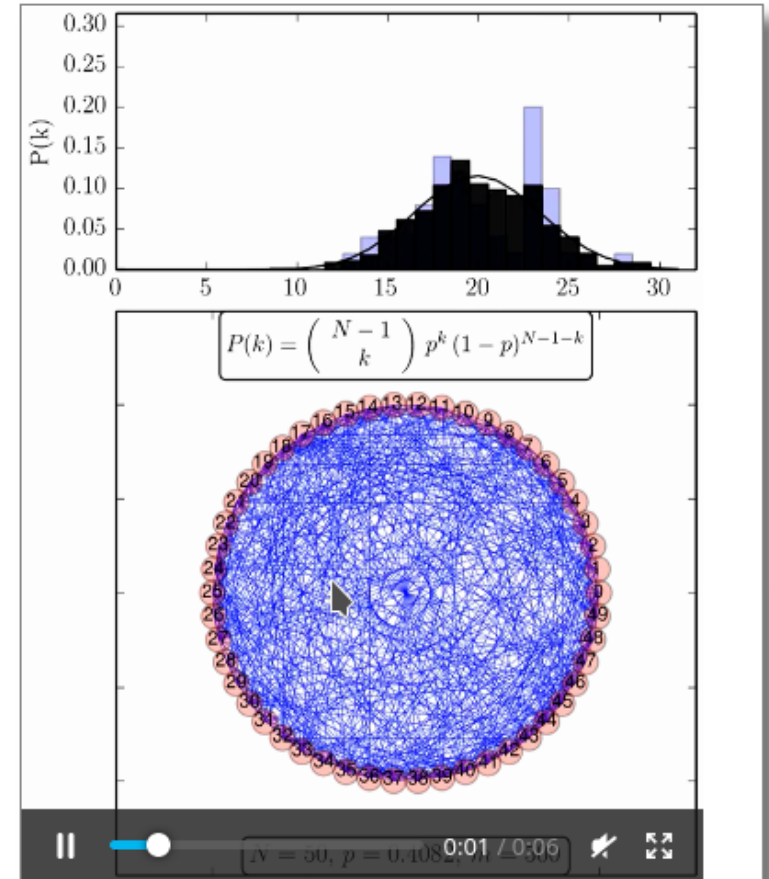
$$P(k) = N(k)/N = \binom{N-1}{k} p^k (1-p)^{N-1-k}$$

Für große  $N$  geht diese Verteilung in die folgende Poisson Verteilungsfunktion über

$$P(k) = \frac{e^{-\bar{k}} \bar{k}^k}{k!},$$

wobei  $\bar{k} = \langle k \rangle = p(N - 1)$  der mittlere Knotengrad im Netzwerk ist (siehe Abb. 3.4 in [Chapter 3: Albert-Laszlo Barabasi, Network Science](#)). Die oberen Abbildungen zeigen Ergebnisse des Python Skriptes ([Python Skript: RandomNetwork.py](#)) welches einen Zufallsgraph mit  $N$ -Knoten und  $m$ -Kanten und die zugehörige Verteilungsfunktion  $N(k) = N P(k)$  der Knotengrade darstellt. Die einzelnen blauen Kästchen des Balkendiagramms stellen die spezifische, zufällig erzeugte Realisierung des Zufallsgraphen dar, wohingegen die schwarze Kurve die analytische Binomialverteilung zeigt. Mittelt man über die einzelnen Verteilungsfunktionen  $P(k)$  mehrerer zufälliger Netzwerke, erzeugt man ein Ensemble-Mittelwert der möglichen Realisierungen des Zufallsgraphen. In der nebenstehenden rechten Abbildung ist eine solche Mittelung anhand einer Animation veranschaulicht. Man erkennt deutlich, dass die über mehrere zufällige Netzwerke gemittelte Verteilung (dunkel graue Kästchen) sich der analytische Verteilung (schwarze Kurve) immer mehr annähert.

Eine weitere wichtige Eigenschaft des Netzwerkes ist die relative Größe des größten verbundenen Knotenclusters (Giant component, Hub)  $N_G/N$ . Zufällige Netzwerke mit kleinem mittlere Knotengrad  $\langle k \rangle < 1$  besitzen keine



Mittelt man über die einzelnen Verteilungsfunktionen  $P(k)$  mehrerer zufälliger Netzwerke so ergeben sich die analytisch ermittelten Verteilungsfunktionen (Python Skript [VPSOC-RandomNetwork\\_evol.py](#)).

# Physik der sozio-ökonomischen Systeme mit dem Computer

## (Physics of Socio-Economic Systems with the Computer)

Vorlesung gehalten an der J.W.Goethe-Universität in Frankfurt am Main

(Sommersemester 2024)

von Dr.phil.nat. Dr.rer.pol. Matthias Hanauske

Frankfurt am Main 01.02.2024

### Jupyter Notebook

Zweiter Vorlesungsteil:

### RandomNetworks.ipynb

Zufällige komplexe Netzwerke (random networks)

### Einführung

Aufgrund ihrer unterschiedlichen Eigenschaften unterscheidet man die folgenden Netzwerk-Klassen: Zufällige Netzwerke (Random Networks: Die einzelnen Kanten bei zufälligen Netzwerke werden von den Knoten (Spielern) nach einem rein zufälligen Muster ausgewählt), Kleine Welt-Netzwerke (Small World Networks, Kleine Welt-Netzwerke zeichnen sich durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten des Netzwerkes und einem großen Wert des Clusterkoeffizienten aus), Exponentielle Netzwerke (Exponential Networks) und Skalenfreie Netzwerke (Scale-Free Networks).

Bei einigen Modellnetzwerken können analytische Ergebnisse gewonnen werden. Im Folgenden betrachten wir die Klasse der zufälligen Netzwerke. Die einzelnen Kanten bei zufälligen Netzwerken werden von den Knoten (Spielern) nach einem rein zufälligen Muster ausgewählt. Im Erdos-Renyi Modell (Erdos and Renyi, 1959) werden N Knoten zufällig mit L ungerichteten Kanten verbunden. Die Wahrscheinlichkeit p, dass ein Knoten mit dem anderen verbunden ist demnach  $p = \frac{2L}{N(N-1)}$ . Die Verteilungsfunktion der Knotengrade  $P(k)$  ist binomialverteilt:

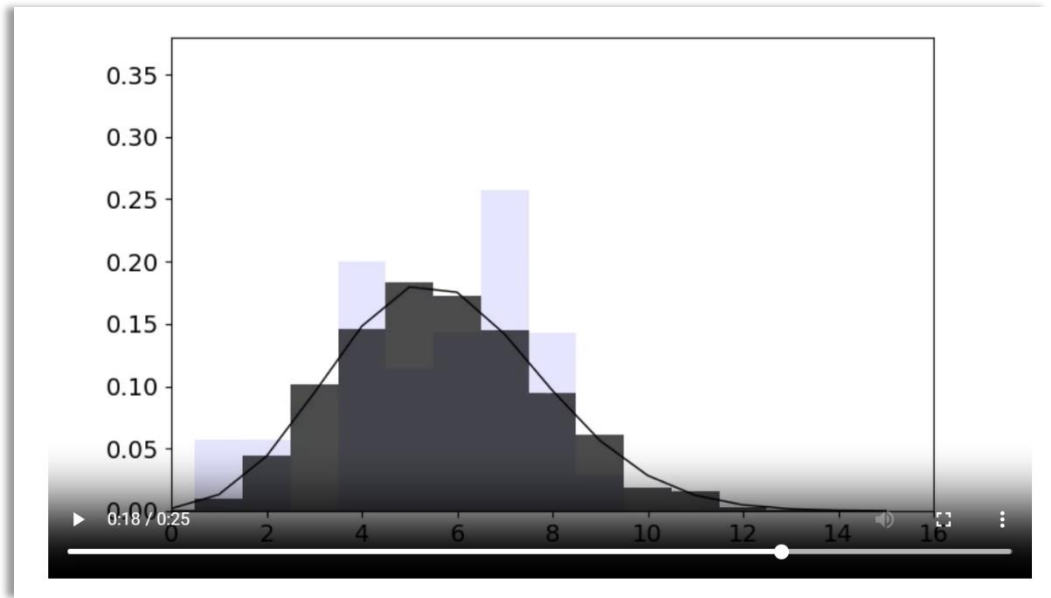
$$P(k) = N(k)/N = \binom{N-1}{k} p^k (1-p)^{N-1-k} \tag{1}$$

Für große N geht diese Verteilung in die folgende Poisson Verteilungsfunktion über

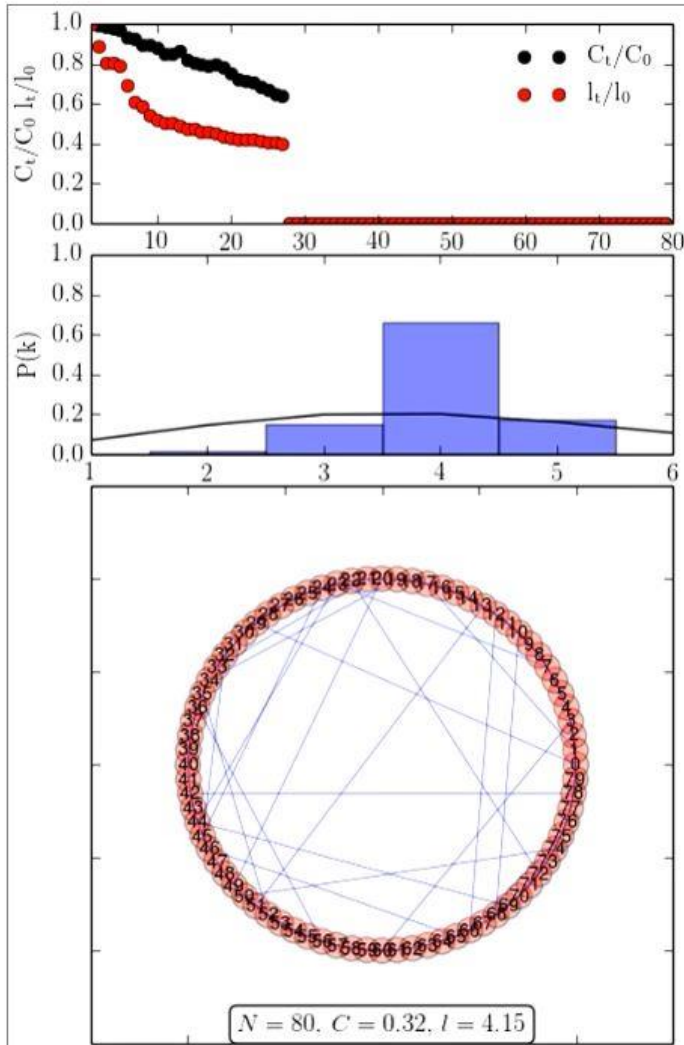
$$P(k) = \frac{e^{-\bar{k}} \bar{k}^k}{k!} , \tag{2}$$

wobei  $\bar{k} = \langle k \rangle = p(N-1)$  der mittlere Knotengrad im Netzwerk ist (siehe Abb. 3.4 in [Chapter 3: Albert-Laszlo Barabasi, Network Science](#)).

Im Folgenden erzeugen wir einen Zufallsgraph mit N-Knoten und L-Kanten



## II.2.2 Kleine Welt Netzwerke (Small-World Networks)



Übergang zwischen einem regulären Netzwerk, einem *Kleine Welt* Netzwerk und einem zufälligen Netzwerk (Python Skript [VPSOC-SmallWorld-Network\\_evol.py](#)).

*Kleine Welt*-Netzwerke zeichnen sich durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten des Netzwerkes und einem großen Wert des Clusterkoeffizienten aus. Ein einfaches Modell welches den Übergang von einem Netzwerk mit regulärer Struktur (Gitter-ähnlicher Struktur) über ein *kleines Welt* Netzwerk hin zu einem zufälligen Netzwerk veranschaulicht, wurde von Watts und Strogatz im Jahre 1998 vorgestellt: Im einfachsten Fall startet man hierbei mit einem eindimensionalen Gitter-Netzwerk mit  $N$  Knoten, wobei jeder Knoten mit seinen  $K$ -nächsten Nachbarn (hier speziell  $K=2$ ) verbunden ist. Nun löscht man mit der Wahrscheinlichkeit  $p$  jede der existierenden Verbindungen (Kanten) und stellt eine neue Kante im Netzwerk in zufälliger Weise her. Für  $p=0$  bleibt die ursprüngliche Gitterstruktur erhalten und für  $p=1$  erzeugt man ein vollständig zufälliges Netzwerk. Watts und Strogatz konnten in ihrem Modell zeigen, dass man schon für kleine  $p$  ( $0 < p \ll 1$ ) ein *kleines Welt*-Netzwerk erzeugt, das durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten und einem großen Wert des Clusterkoeffizienten gekennzeichnet ist.

Die linke, nebenstehende Abbildung zeigt eine Animation eines leicht abgeänderten Watts-Strogatz Modells. Zum Zeitpunkt  $t=0$  startet man wiederum mit einer regulären Gitterstruktur (hier  $N=80$ ,  $K=2$ ). Zu jedem folgenden Zeitpunkt wird nun zufällig eine Kante im Netzwerk ausgewählt, die in zufälliger Weise neu angeordnet wird. Im Laufe der Zeitentwicklung erkennt man, dass sich die Verteilungsfunktion der Knotengrade immer mehr an die binomialverteilte Funktion des zufälligen Netzwerkes annähert ( $P(k)$ , siehe schwarze Kurve im mittleren Diagramm in der linken Animation). Im oberen Diagramm der linken Animation ist hingegen der normierte Wert des Clusterkoeffizienten ( $C_t/C_0$ , schwarze Punkte) und der normierte Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten des Netzwerkes ( $l_t/l_0$ , rote Punkte) als Funktion des Iterationszeitpunktes  $t$  dargestellt. Man erkennt, dass sich  $l_t$  schon nach wenigen Iterationsschritten schnell verkleinert - im Bereich  $5 < t < 20$  hat das Netzwerk eine klare *Kleine Welt*-Struktur. Für große Zeiten nähert sich das Netzwerk immer weiter einem zufälligen Netzwerk an.

Neben den hier dargestellten Netzwerkeigenschaften von zufälligen und kleinen Welt Netzwerken findet sich eine ausführliche Darstellung in [Chapter 3: Albert-Laszlo Barabasi, Network Science](#). In der Section 3

# Physik der sozio-ökonomischen Systeme mit dem Computer

## (Physics of Socio-Economic Systems with the Computer)

Vorlesung gehalten an der J.W.Goethe-Universität in Frankfurt am Main

(Sommersemester 2024)

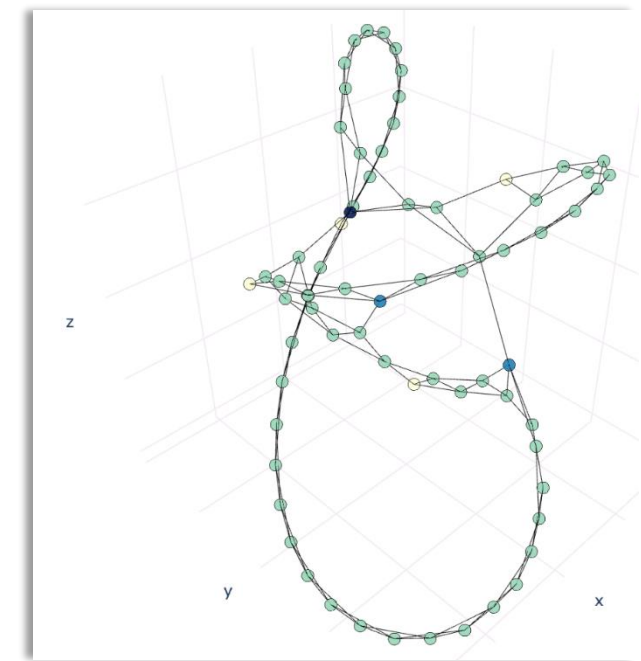
von Dr.phil.nat. Dr.rer.pol. Matthias Hanauske

Frankfurt am Main 01.02.2024

Zweiter Vorlesungsteil:

Kleine Welt Netzwerke (small world networks)

## Jupyter Notebook SmallWorldNetworks.ipynb



### Einführung

Aufgrund ihrer unterschiedlichen Eigenschaften unterscheidet man die folgenden Netzwerk-Klassen: Zufällige Netzwerke (Random Networks: Die einzelnen Kanten bei zufälligen Netzwerken werden von den Knoten (Spielern) nach einem rein zufälligen Muster ausgewählt), Kleine Welt-Netzwerke (Small World Networks, Kleine Welt-Netzwerke zeichnen sich durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten des Netzwerkes und einem großen Wert des Clusterkoeffizienten aus), Exponentielle Netzwerke (Exponential Networks) und Skalenfreie Netzwerke (Scale-Free Networks). Bei einigen Modellnetzwerken können analytische Ergebnisse gewonnen werden. Im Folgenden betrachten wir die Klasse der kleinen Welt Netzwerke.

"Kleine Welt"-Netzwerke zeichnen sich durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten des Netzwerkes und einem großen Wert des Clusterkoeffizienten aus. Ein einfaches Modell, welches den Übergang von einem Netzwerk mit regulärer Struktur (Gitter-ähnlicher Struktur) über ein "kleines Welt"-Netzwerk hin zu einem zufälligen Netzwerk veranschaulicht, wurde von Watts und Strogatz im Jahre 1998 vorgestellt. Im einfachsten Fall startet man hierbei mit einem eindimensionalen Gitter-Netzwerk mit  $N$  Knoten, wobei jeder Knoten mit seinen  $K$ -nächsten Nachbarn (im folgenden Beispiel speziell  $K=2$ ) verbunden ist. Nun löscht man mit der Wahrscheinlichkeit  $p$  jede der existierenden Verbindungen (Kanten) und stellt eine neue Kante im Netzwerk in zufälliger Weise her. Für  $p=0$  bleibt die ursprüngliche Gitterstruktur erhalten und für  $p=1$  erzeugt man ein vollständig verbundenes Netzwerk. Watts und Strogatz konnten in ihrem Modell zeigen, dass man schon für kleine  $p$  ( $0 < p \ll 1$ ) ein "kleines Welt"-Netzwerk erzeugt, das durch einen kleinen Wert der durchschnittlichen kürzesten Verbindung zwischen den Knoten und einem großen Wert des Clusterkoeffizienten gekennzeichnet ist. Im Folgenden veranschaulichen wir das Prinzip eines Watts-Strogatz "kleine Welt"-Netzwerks.

Wir erstellen zunächst ein reguläres Ring-Netzwerk, wobei jeder Knoten mit seinem nächsten und übernächsten Nachbarn verbunden ist.

```
import networkx as nx
from random import randint
```

# Network Science

by Albert-László Barabási

Personal Introduction

1. Introduction

2. Graph Theory

3. Random Networks

4. The Scale-Free Property

5. The Barabási-Albert Model

6. Evolving Networks

7. Degree Correlations

8. Network Robustness

9. Communities

10. Spreading Phenomena

Preface

<http://networksciencebook.com/>

Start Reading

English

Русский

Magyar

فارسی

日本語

### Section 3.2

#### The Random Network Model

Wie erzeugt man mittels eines mathematischen Algorithmus ein zufälliges Netzwerk (siehe Box 3.1)

### Section 3.4

#### Degree Distribution

Wie sieht die Verteilungsfunktion der Knotengrade in zufällige Netzwerken aus (siehe Image 3.4 Binomial vs. Poisson Degree Distribution)

### Section 3.5

#### Real Networks are Not Poisson

Vergleich: Real existierende Netzwerke  $\leftrightarrow$  Zufällige Netzwerke (siehe Image 3.6 Degree Distribution of Real Networks)

### Section 3.6

#### The Evolution of a Random Network

Relativen Größe des Hubs (grösster verbundener Knotencluster) hängt von dem durchschnittlichen Knotengrad des Netzwerkes ab.  
Definition von unterschiedlichen Regimen in zufälligen Netzwerken (subcritical, supercritical, fully connected) (siehe Image 3.7 Evolution of a Random Network)

### Section 3.7

#### Real Networks are Supercritical

Sind real existierende Netzwerke subcritical, supercritical oder fully connected? (siehe Table 3.1 Are Real Networks Connected? und Image 3.9 Most Real Networks are Supercritical)

### Section 3.8

#### Small Worlds

Definition der kleinen Welt Eigenschaft in komplexen Netzwerken "In the language of network science the small world phenomenon implies that the distance between two randomly chosen nodes in a network is short." Mittlerer Abstand zwischen zwei Knoten im Netzwerk  $\langle d \rangle$  bestimmt die Eigenschaft von kleinen Welt Netzwerken (siehe Image 3.10 Six Degree of Separation and Image 3.11 Why are Small Worlds Surprising? und Table 3.2 Six Degrees of Separation)

### Section 3.9

#### Clustering Coefficient

Der Clusterkoeffizient in real existierenden und zufälligen Netzwerken (siehe Image 3.13 Clustering in Real Networks und Box 3.9 Watts-Strogatz Model)

## Random Networks: a Brief History

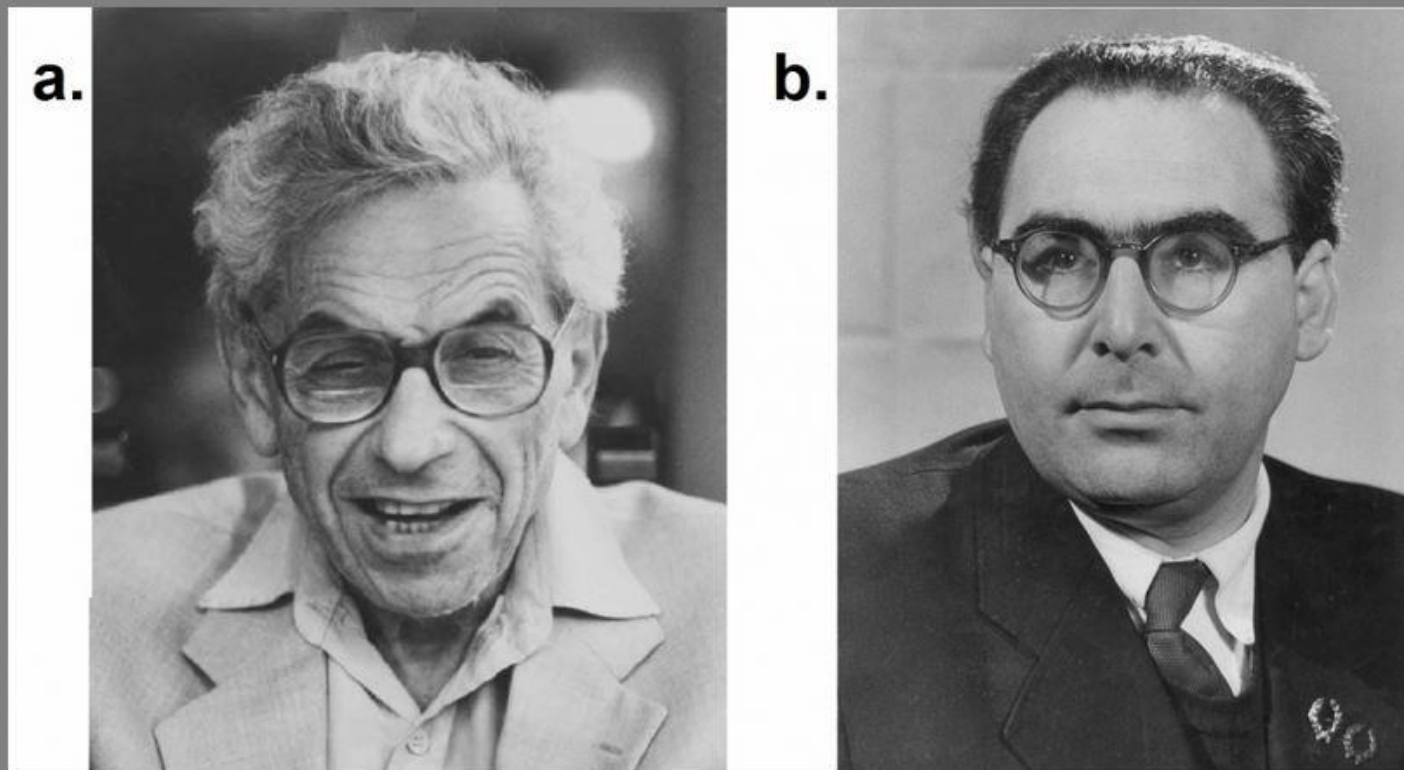


Image 3.2

**(a) Pál Erdős (1913-1996)**

Hungarian mathematician known for both his exceptional scientific output and eccentricity. Indeed, Erdős published more papers than any other mathematician in the history of mathematics. He co-authored papers with over five hundred mathematicians, inspiring the concept of *Erdős number*. His legendary personality and profound professional impact has inspired two biographies [12, 13] and a documentary [14] ([Video 3.1](#)).

**(b) Alfréd Rényi (1921-1970)**

Hungarian mathematician with fundamental contributions to combinatorics, graph theory, and number theory. His impact goes beyond mathematics: The Rényi entropy is widely used in chaos theory and the random network theory he co-developed is at the heart of network science. He is remembered through the hotbed of Hungarian mathematics, the Alfréd Rényi Institute of Mathematics in Budapest.



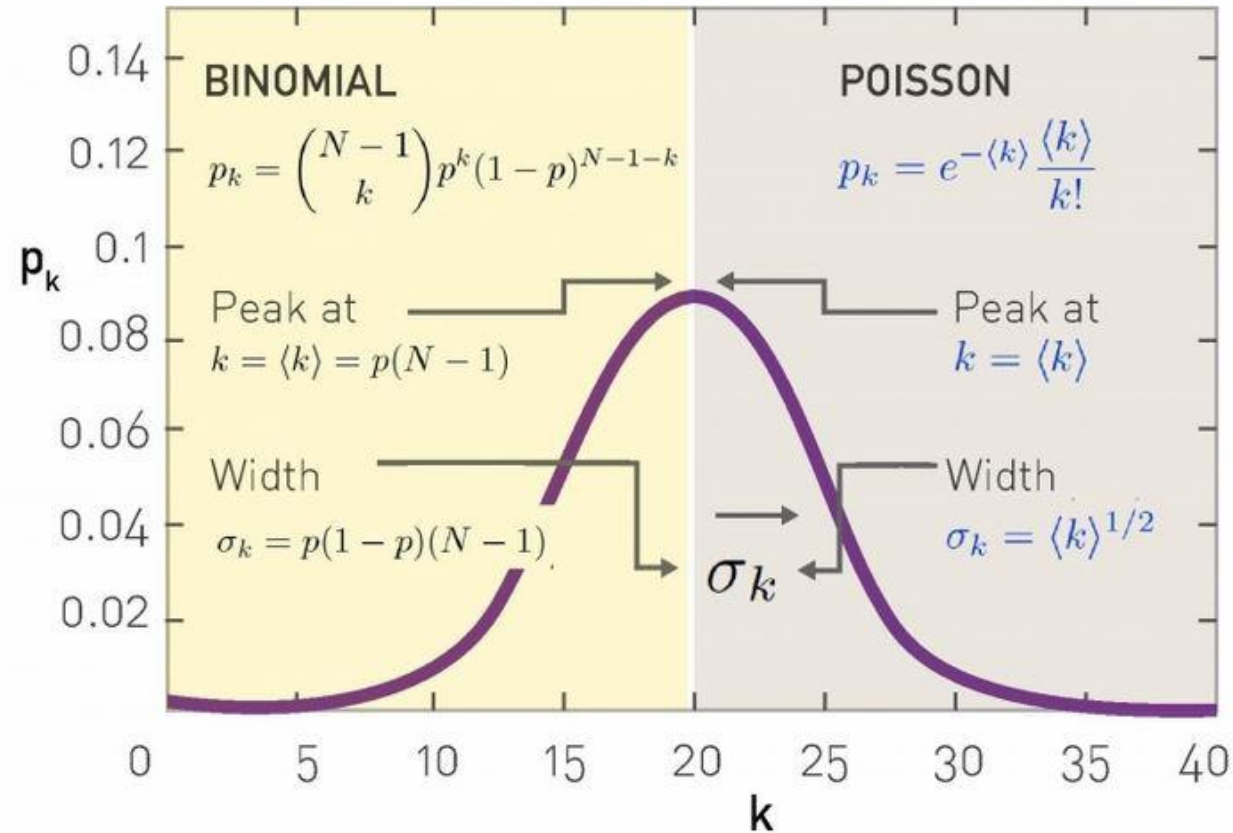


Image 3.4

### Binomial vs. Poisson Degree Distribution

The exact form of the degree distribution of a random network is the binomial distribution (left half). For  $N \gg \langle k \rangle$  the binomial is well approximated by a Poisson distribution (right half). As both formulas describe the same distribution, they have the identical properties, but they are expressed in terms of different parameters: The binomial distribution depends on  $p$  and  $N$ , while the Poisson distribution has only one parameter,  $\langle k \rangle$ . It is this simplicity that makes the Poisson form preferred in calculations.

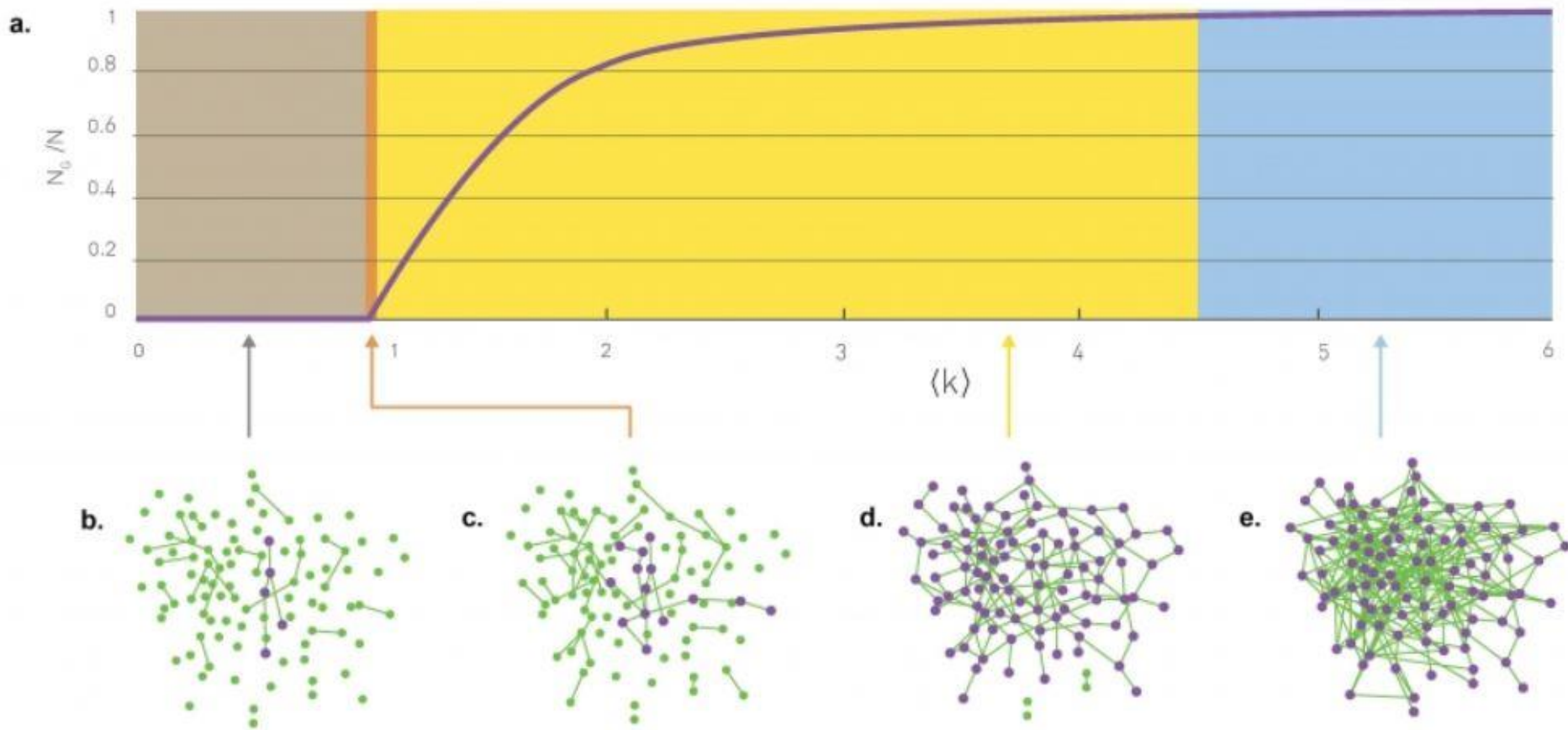


Image 3.7

### Evolution of a Random Network

- The relative size of the giant component in function of the average degree  $\langle k \rangle$  in the Erdős-Rényi model. The figure illustrates the phase transition at  $\langle k \rangle = 1$ , responsible for the emergence of a giant component with nonzero  $N_G$
- A sample network and its properties in the four regimes that characterize a random network.

## Small Worlds

The *small world phenomenon*, also known as *six degrees of separation*, has long fascinated the general public. It states that if you choose any two individuals anywhere on Earth, you will find a path of at most six acquaintances between them ([Image 3.10](#)). The fact that individuals who live in the same city are only a few handshakes from each other is by no means surprising. The small world concept states, however, that even individuals who are on the opposite side of the globe can be connected to us via a few acquaintances.

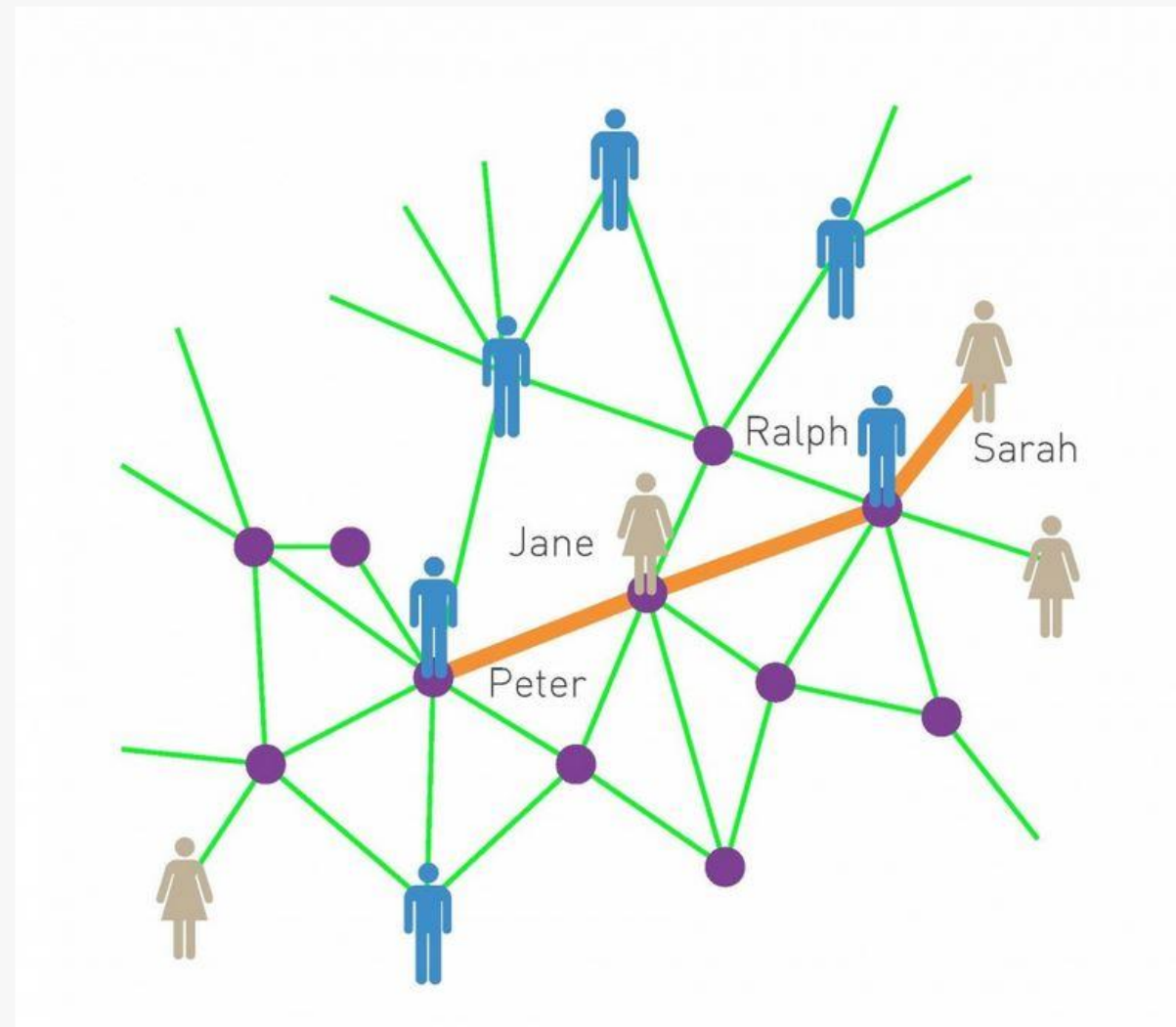
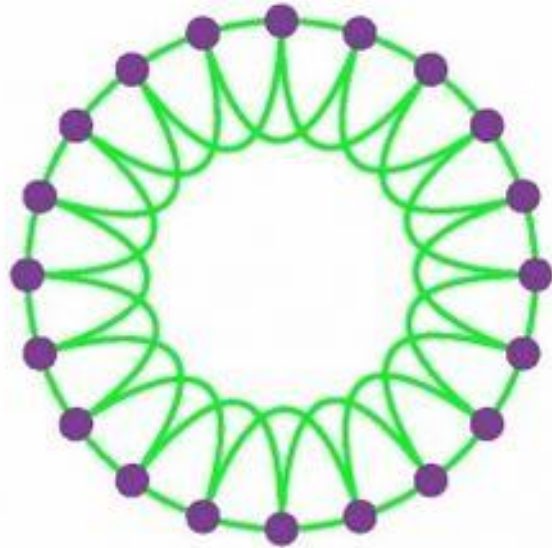


Image 3.10

Six Degree of Separation

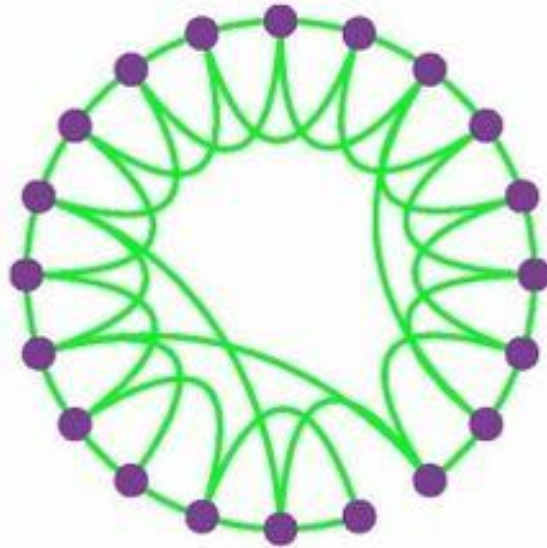
**a.**

REGULAR



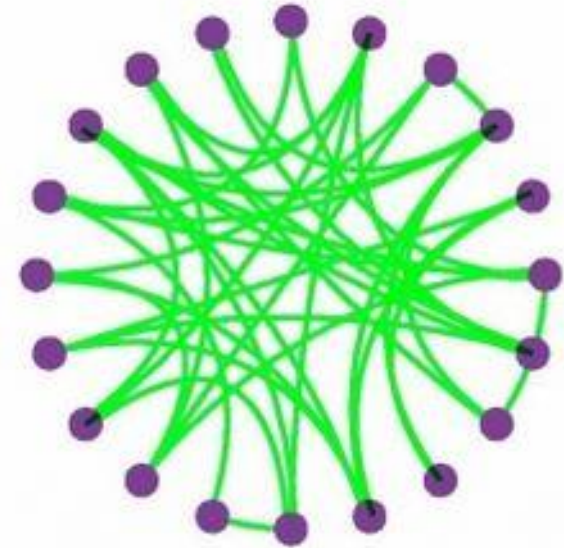
**b.**

SMALL-WORLD



**c.**

RANDOM

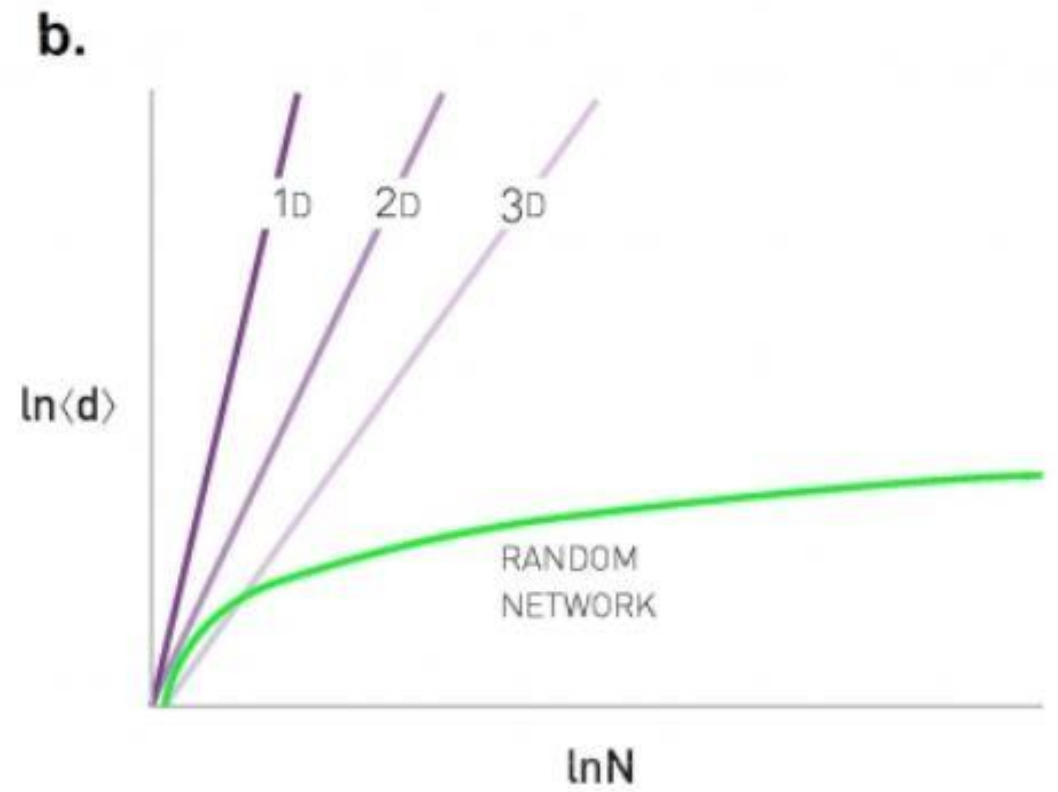
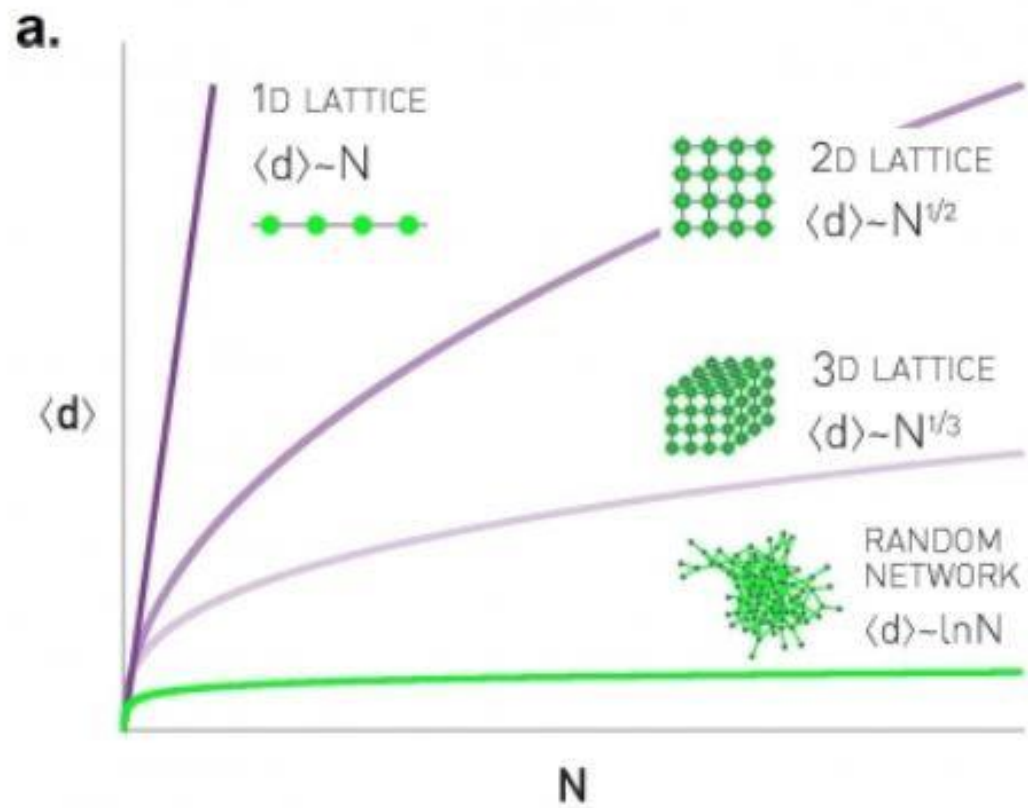


$p = 0$



$p = 1$

Increasing randomness



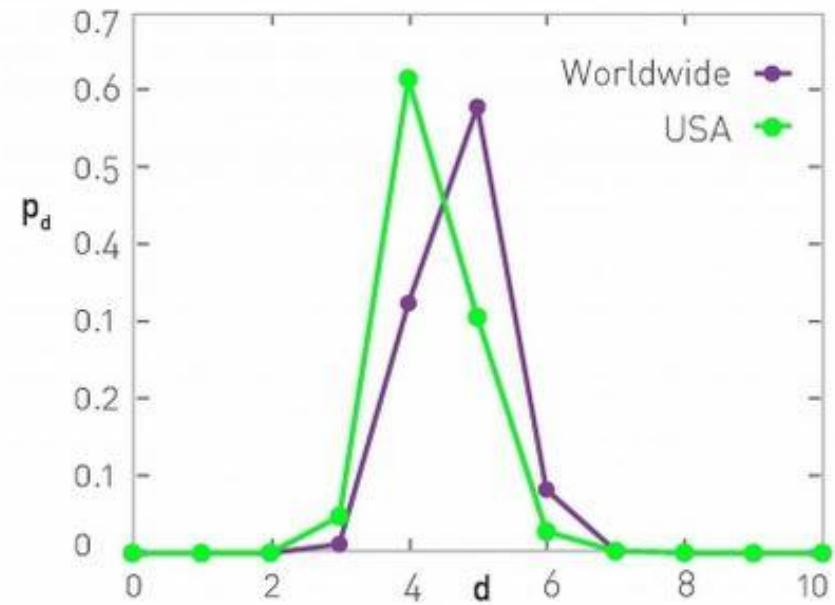
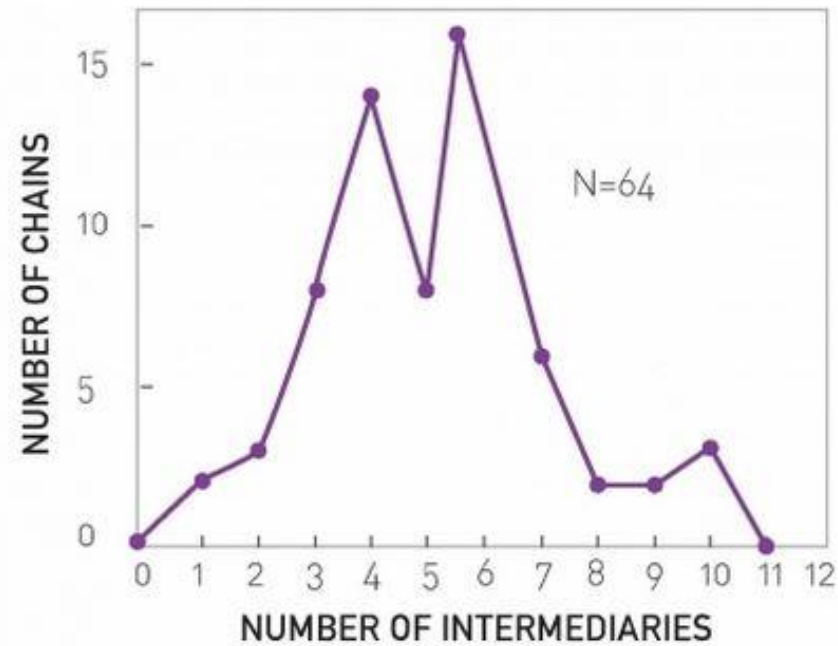


Image 3.12

### Six Degrees? From Milgram to Facebook

- In Milgram's experiment 64 of the 296 letters made it to the recipient. The figure shows the length distribution of the completed chains, indicating that some letters required only one intermediary, while others required as many as ten. The mean of the distribution was 5.2, indicating that on average six 'handshakes' were required to get a letter to its recipient. The playwright John Guare renamed this 'six degrees of separation' two decades later. After [25].
- The distance distribution,  $p_d$ , for all pairs of Facebook users worldwide and within the US only. Using Facebook's  $N$  and  $L$  (3.19) predicts the average degree to be approximately 3.90, not far from the reported four degrees. After [18].

## 19 Degrees of Separation

How many clicks do we need to reach a randomly chosen document on the Web? The difficulty in addressing this question is rooted in the fact that we lack a complete map of the WWW—we only have access to small samples of the full map. We can start, however, by measuring the WWW's average path length in samples of increasing sizes, a procedure called *finite size scaling*. The measurements indicate that the average path length of the WWW increases with the size of the network as [21]

$$\langle d \rangle \approx 0.35 + 0.89 \ln N$$

In 1999 the WWW was estimated to have about 800 million documents [22], in which case the above equation predicts  $\langle d \rangle \approx 18.69$ . In other words in 1999 two randomly chosen documents were on average 19 clicks from each other, a result that became known as *19 degrees of separation*. Subsequent measurements on a sample of 200 million documents found  $\langle d \rangle \approx 16$  [23], in good agreement with the  $\langle d \rangle \approx 17$  prediction. Currently the WWW is estimated to have about trillion nodes ( $N \sim 10^{12}$ ), in which case the formula predicts  $\langle d \rangle \approx 25$ . Hence  $\langle d \rangle$  is not fixed but as the network grows, so does the distance between two documents.

The average path length of 25 is much larger than the proverbial six degrees (BOX 3.7). The difference is easy to understand: The WWW has smaller average degree and larger size than the social network. According to (3.19) both of these differences increase the Web's diameter.

Network	N	L	$\langle k \rangle$	$\langle d \rangle$	$d_{max}$	$\ln N / \ln \langle k \rangle$
Internet	192,244	609,066	6.34	6.98	26	6.58
WWW	325,729	1,497,134	4.60	11.27	93	8.31
Power Grid	4,941	6,594	2.67	18.99	46	8.66
Mobile-Phone Calls	36,595	91,826	2.51	11.72	39	11.42
Email	57,194	103,731	1.81	5.88	18	18.4
Science Collaboration	23,133	93,437	8.08	5.35	15	4.81
Actor Network	702,388	29,397,908	83.71	3.91	14	3.04
Citation Network	449,673	4,707,958	10.43	11.21	42	5.55
E. Coli Metabolism	1,039	5,802	5.58	2.98	8	4.04
Protein Interactions	2,018	2,930	2.90	5.61	14	7.14

Table 3.2

### Six Degrees of Separation

The average distance  $\langle d \rangle$  and the maximum distance  $d_{max}$  for the ten reference networks. The last column provides  $\langle d \rangle$  predicted by (3.19), indicating that it offers a reasonable approximation to the measured  $\langle d \rangle$ . Yet, the agreement is not perfect - we will see in the next chapter that for many real networks (3.19) needs to be adjusted. For directed networks the average degree and the path lengths are measured along the direction of the links.



## Weiterführende Links

- [Folien der 6.Vorlesung](#)
- [Vorlesungsaufzeichnung der 6.Vorlesung: WS 2022/23 bzw. WS 2021/22](#)
- [View Jupyter Notebook: Einführung in die Theorie der komplexe Netzwerke](#)
- [Download Jupyter Notebook: Einführung in die Theorie der komplexe Netzwerke](#)
- [View Jupyter Notebook: Zufällige komplexe Netzwerke \(random networks\)](#)
- [Download Jupyter Notebook: Zufällige komplexe Netzwerke \(random networks\)](#)
- [View Jupyter Notebook: Kleine Welt Netzwerke \(small world networks\)](#)
- [Download Jupyter Notebook: Kleine Welt Netzwerke \(small world networks\)](#)
- [Download Python Programm: Zufälliges komplexes Netzwerk](#)
- [Download Python Programm: Mittelung mehrerer zufälliger komplexer Netzwerke](#)
- [Download Python Programm: Kleine Welt Netzwerke \(Watts-Strogatz Modell\)](#)