

## Modellbildungssysteme: Pädagogische und didaktische Ziele

Was hat Modellbildung mit der Schule zu tun? Der Bildungsplan 1994 formuliert:

"Die schnelle Zunahme des Wissens, die hohe Differenzierung und die komplexen Strukturen in allen Bereichen der Gesellschaft, Wirtschaft und Technik erfordern in zunehmenden Maße übergreifendes Denken in Zusammenhängen."

Genau dieser Forderung genügt das Arbeiten mit einem Modellbildungssystem. Der Anwender muss wissen, welche Zusammenhänge zwischen den einzelnen Objekten in seiner Simulation bestehen. Glaubt er diese zu kennen, so kann er in sein Modellbildungssystem ein Modell eingeben und es dann simulieren. Durch Vergleich der Simulationsergebnisse mit dem Experiment kann der Anwender seine Theorie überprüfen und gegebenenfalls korrigieren. Ein Modellbildungssystem verlangt und fördert demnach das Denken in Zusammenhängen.

Der Einsatz von Modellbildungssystemen fördert das Verständnis sowohl der differentiellen als auch der integralen Betrachtungsweise.

Beispielsweise ist die Aktivität eines Stoffes ein Maß dafür, wie schnell sich die Stoffmenge verändert. Bei näherer Betrachtung erkennt man, dass die Aktivität die (negative) zeitliche Ableitung der Stoffmenge ist.

### Beispiel "Der radioaktive Zerfall von Radonkernen"

Ich möchte anhand des Radonzerfalls einen möglichen Unterrichtsgang zum Thema „Anwendung von Modellbildungssystemen im Physikunterricht“ vorstellen.

#### Arbeitsweise eines Modellbildungssystems

Schauen wir uns zunächst die Stoffmenge an Radonkernen an. Gehen wir davon aus, dass nach einer Sekunde von dieser Stoffmenge (z.B. 10000 Kerne) eine bestimmte Anzahl an Kernen (z.B. 100) zerfallen ist.

Wie viele Kerne sind nach einer Sekunde zerfallen, wenn wir  $2 \cdot 10000$  Kerne betrachten?

Klar, doppelt so viele wie bei 10000 Kernen. Die Anzahl der Zerfälle pro Sekunde ist also proportional zur Stoffmenge ( $R$ ). Die zugehörige Proportionalitätskonstante (Zerfallskonstante) bezeichnen wir mit  $ZK$ .

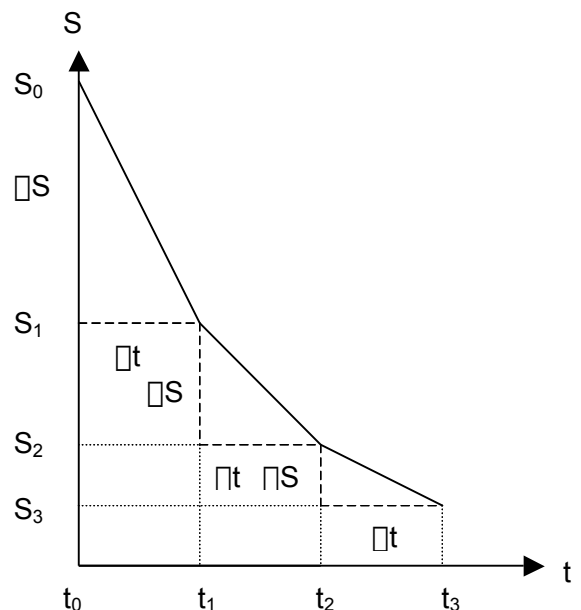
Mit unseren Zahlen wäre  $ZK = \frac{100}{10000} = 0,01$ .

Ändert sich die Stoffmenge  $R$ , so ändern sich auch die Radonzerfälle pro Sekunde (die Aktivität). Betrachten wir unsere Stoffmenge beispielsweise über eine Minute, so hat sich die Stoffmenge und damit auch ihre Aktivität enorm geändert. Wollen wir trotzdem die Stoffmenge nach einer Minute ermitteln, so müssen wir für jede Sekunde die Aktivität berechnen und aus ihr die Stoffmenge der nächsten Sekunde.

Wiederholen wir dieses Verfahren mit  $\Delta t = 1$  Sekunde, so "hangeln" wir uns immer näher an den gesuchten Wert von  $S$  heran. Dieses Verfahren wird auch Euler-Verfahren genannt.

$$S_{\text{neu}} = S_{\text{alt}} + \Delta S = S_{\text{alt}} + ZK \cdot S_{\text{alt}} \cdot \Delta t$$

Damit das Verfahren läuft, muss am Anfang (zum Zeitpunkt  $t_0$ ) der Startwert  $S_0$  bekannt sein.



## Umsetzung mithilfe von Dynasys

### Das erste Modell



In der Modellbildung werden solche Größen **Bestandsgrößen** genannt. Bestandsgrößen können niemals direkt, sondern immer nur indirekt mit sogenannten **Änderungsraten**, bei uns die Aktivität, berechnet werden.

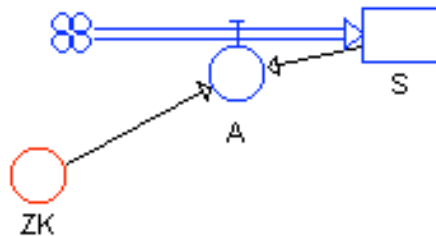
In unserem Beispiel ist die Änderungsrate gleich  $ZK \cdot S$ .

Konstanten, wie unsere Proportionalitätskonstante  $ZK$  oder berechenbare Größen, werden mit dem **Größensymbol** in der Modellbildung dargestellt.

Jetzt haben wir fast alles, was wir brauchen. Wir müssen in unserem Modell nur noch darstellen, dass für die Berechnung der Änderungsrate (der Aktivität)  $A$  von  $S$ , die Größen  $ZK$  und  $S$  selbst benötigt werden. Hierfür werden sogenannte **Wirkungspfeile** eingesetzt. Sie veranschaulichen, welche Größe auf welche eine Wirkung ausübt.

Unser Modell könnte so aussehen:

Die Aktivität ist hier negativ gewählt. Das liegt daran, dass die zerfallenen Kerne nicht zu  $S$  hinzugezählt werden dürfen, sondern abgezogen werden müssen.

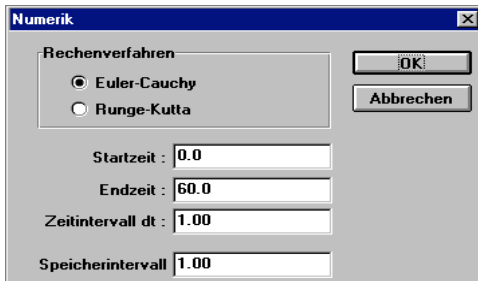


Startwert  $S = 10000$

Zustandsänderungen  
 $A = -ZK \cdot S$

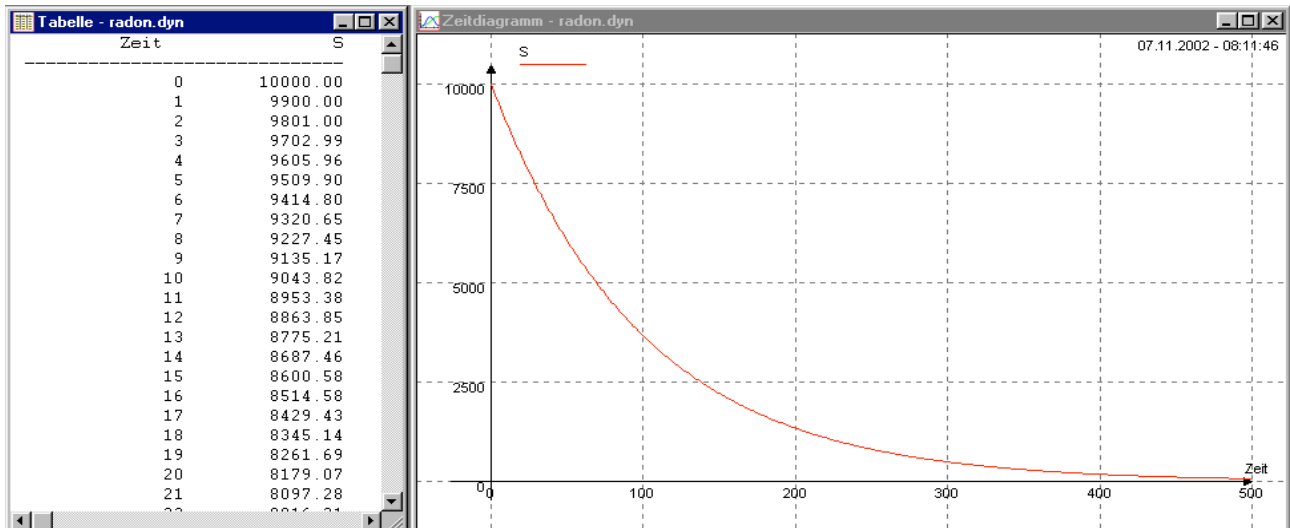
Konstanten  
 $ZK = 0.01$

### Die Simulation



Nachdem das Modell fertig gestellt ist, kann die Simulation vorbereitet werden. Dazu muss man die Start- und Endzeit, die Schrittweite der Simulation  $t$  und das Simulationsverfahren angeben. Das Euler-Verfahren kennen wir bereits. Auf das Runge-Kutte-Verfahren (Runge-Kutta4) wird später noch genauer eingegangen.

Die Simulation kann jetzt gestartet werden. Man kann sich das Simulationsergebnis als Schaubild oder in Form einer Tabelle ausgeben lassen.





Das Schaubild zeigt den exponentiellen Zerfall der Radonkerne.

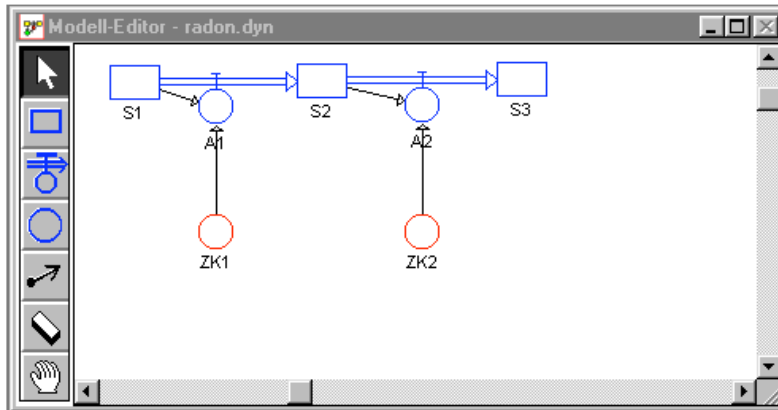
### Erweiterung auf die Simulation einer Zerfallskette

Nun haben wir den Stoff  $S_1$ . Sein Zerfallsprodukt ist  $S_2$  dessen Zerfallsprodukt  $S_3$  ist.  $S_3$  selbst ist stabil. Die Zerfallskonstanten sind entsprechend  $ZK_1$  und  $ZK_2$ .

Wenn wir uns das Änderungsratsymbol etwas genauer betrachten, so fällt auf, dass es aus einem vordern und einem hinteren Teil besteht:

 stellt einen Abfluss dar. Und  einen Zufluss. Wenn wir also die Bestandsgrößen  $S_1$

und  $S_2$  mit einer Änderungsrate verbinden, so werden die Kerne, die von  $S_1$  abgezogen werden zu  $S_2$  hinzugezählt. Auf diese Weise lassen sich sehr anschauliche Modelle erstellen:



Zustandsgleichungen

```
S1.neu <-- S1.alt + dt*(-A1)
Startwert S1 = 10000
S2.neu <-- S2.alt + dt*(A1-A2)
Startwert S2 = 0
S3.neu <-- S3.alt + dt*(A2)
Startwert S3 = 0
```

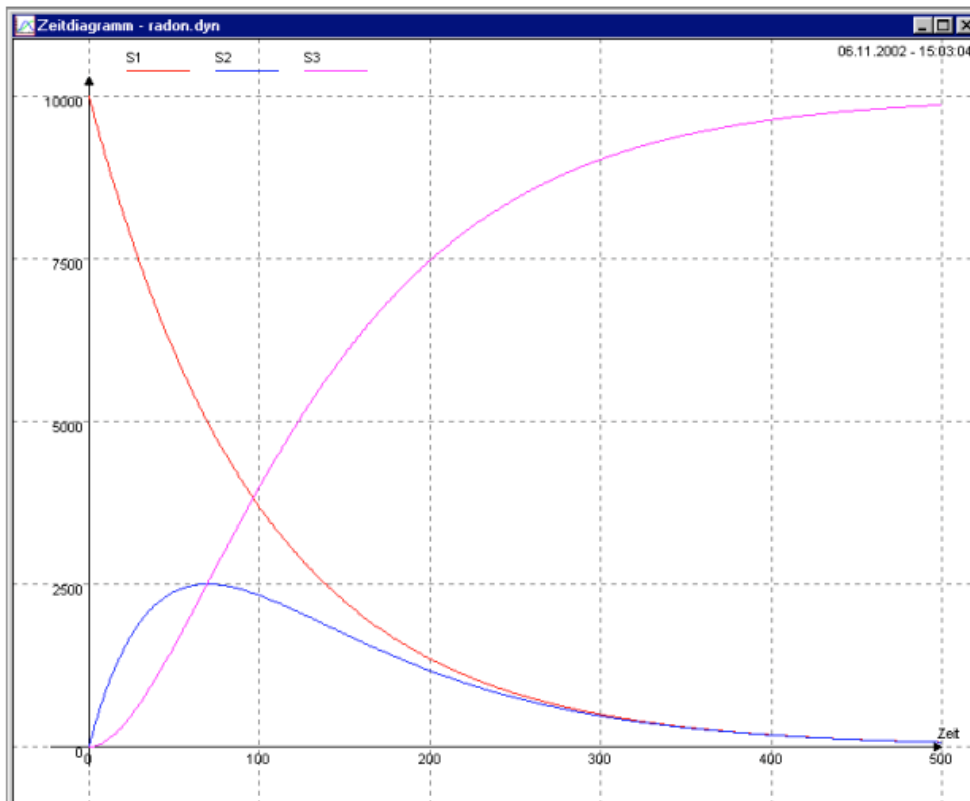
Zustandsänderungen

```
A1 = ZK1*S1
A2 = ZK2*S2
```

Konstanten

```
ZK1 = 0.01
ZK2 = 0.02
```

Zwischenwerte



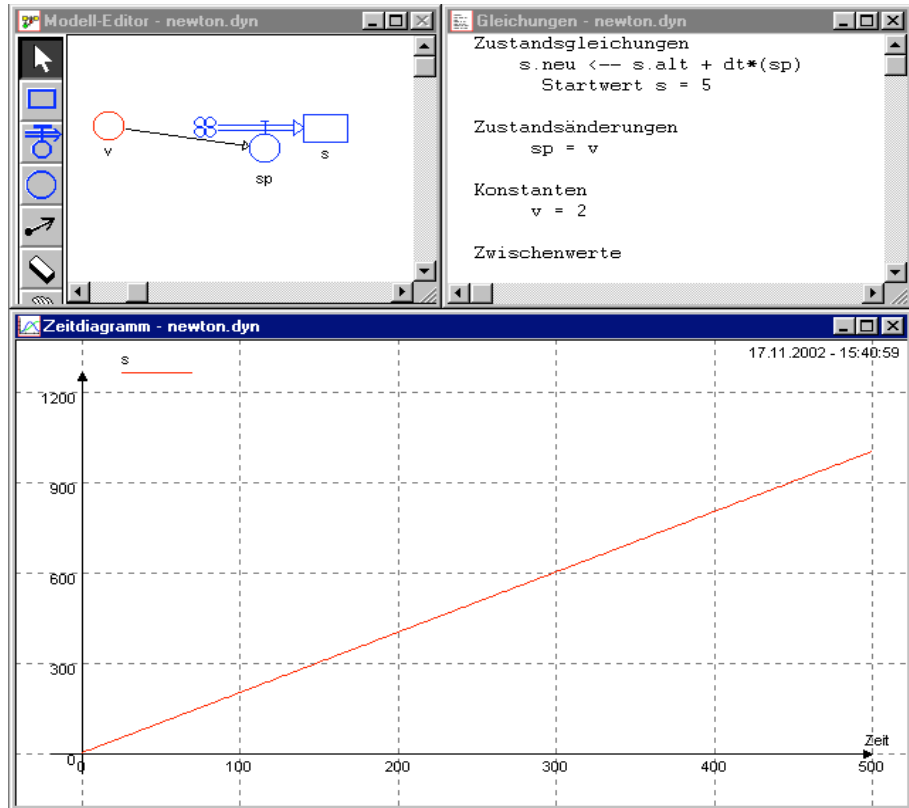
### Die Newton-Maschine

Mit Newton-Maschine bezeichne ich ein Modell, das die Newton'schen Bewegungsgleichungen darstellt. Gehen wir zunächst von einer konstanten Geschwindigkeit aus. Dann gilt für s:

$$s_{\text{neu}} = s_{\text{alt}} + v \cdot \Delta t \text{ oder } v = \frac{s_{\text{neu}} - s_{\text{alt}}}{\Delta t} \text{ mit } \Delta t = t_{\text{neu}} - t_{\text{alt}} .$$

Man sieht, falls  $\Delta t \rightarrow 0$  geht, ist der Grenzwert von  $\frac{s_{\text{neu}} - s_{\text{alt}}}{\Delta t}$  die zeitliche Ableitung von s.

Das Modell könnte so aussehen:



Im nächsten Schritt gehen wir von einer konstanten Beschleunigung aus.

Entsprechend sehen wir:

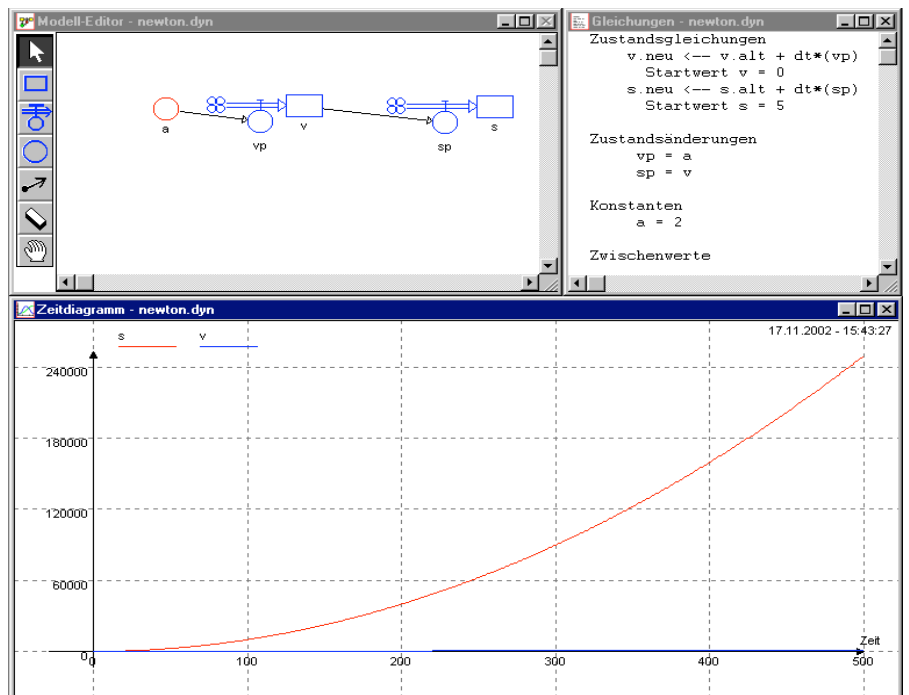
$$v_{\text{neu}} = v_{\text{alt}} + a \cdot \Delta t$$

$$\text{oder } a = \frac{v_{\text{neu}} - v_{\text{alt}}}{\Delta t}$$

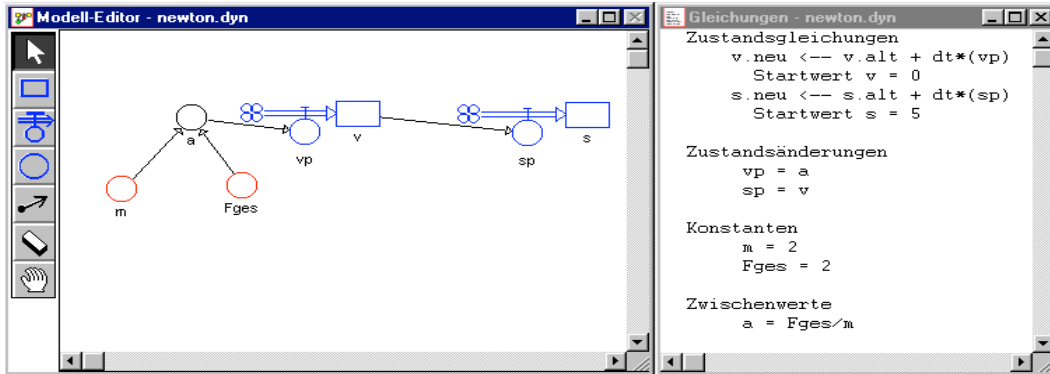
und somit ist

$$a = \dot{v} \text{ für } \Delta t \rightarrow 0 .$$

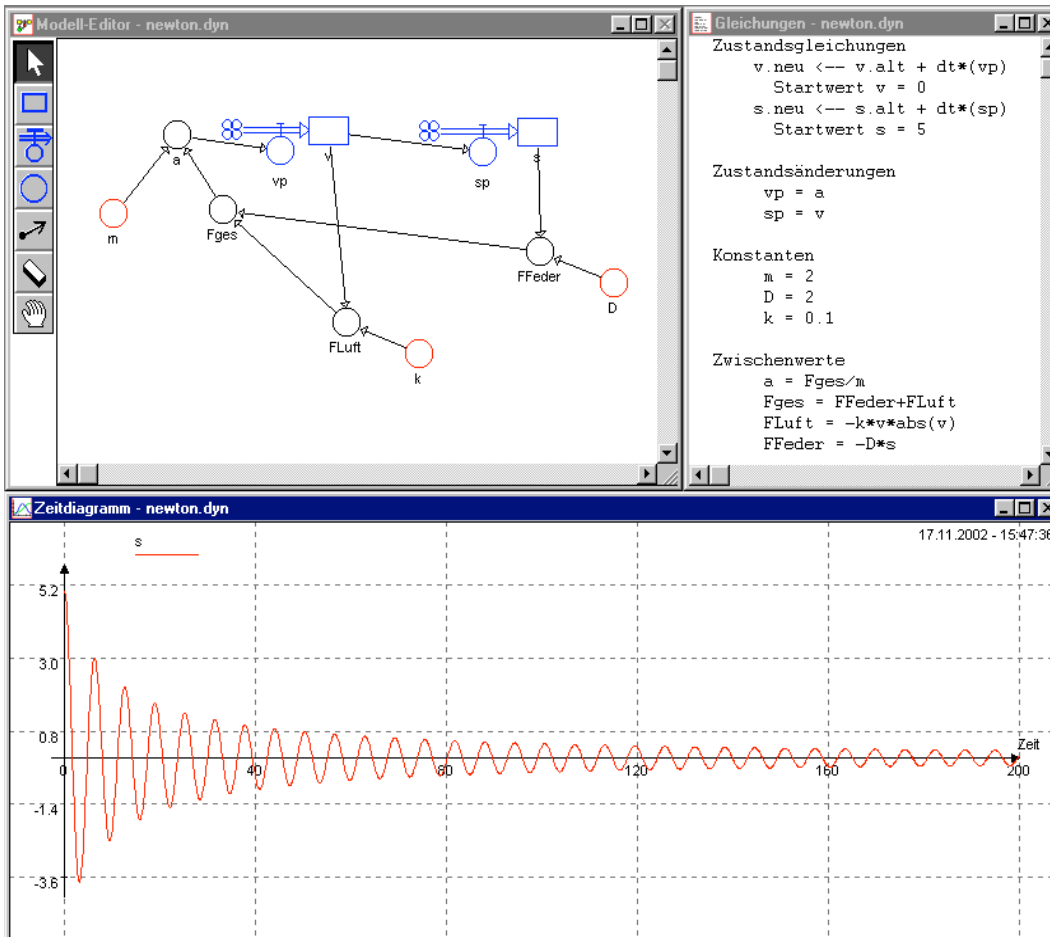
Mit dieser Erweiterung erhält man nebenstehendes Modell.



Nach Newton ist  $a = \frac{F}{m}$ . Das folgende Modell ist damit ein Modell der Newton'schen Bewegungsgleichungen, das wir Newtonmaschine nennen wollen:



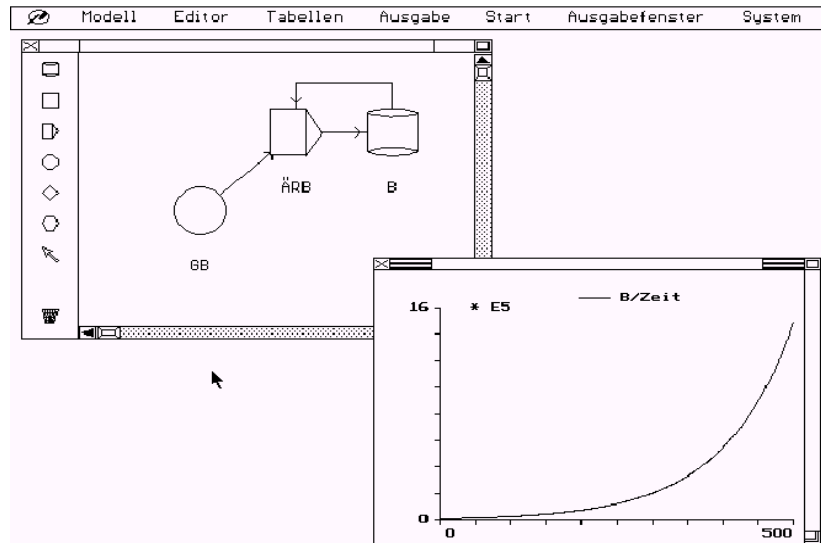
Mit ihr lassen sich auch sehr komplexe Bewegungsvorgänge simulieren. Wir müssen lediglich die verschiedenen Kräfte berechnen und zu einer Gesamtkraft addieren. Das folgende Bild zeigt ein Modell zur Simulation einer Federschwingung mit Luftwiderstand.



## Verschiedene Modellbildungssysteme im Vergleich

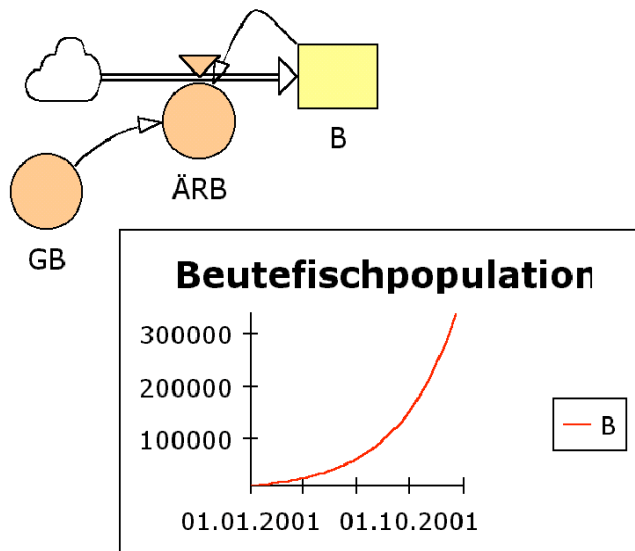
Ich habe die Modellbildungssysteme MODUS, Powersim, Stella, Moebius und Dynasys getestet. Powersim, Stella und Dynasys sind professionelle Windows-Programme.

### Modus



MODUS ist das einzige Modellbildungssystem, das für die Ausbildung entwickelt wurde. Leider wurde es in den letzten Jahren nicht weiter entwickelt. MODUS ist ein DOS-Programm, das nur einen Bruchteil der heutigen Rechnerleistung ausnutzt. Trotzdem hat es gegenüber den anderen professionellen Modellbildungssystemen Vorteile. Es hat eine deutsche Benutzerführung und es leistet nur das, was man in der Schule benötigt.

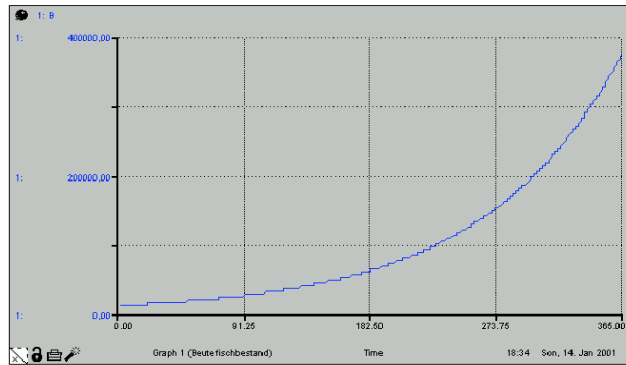
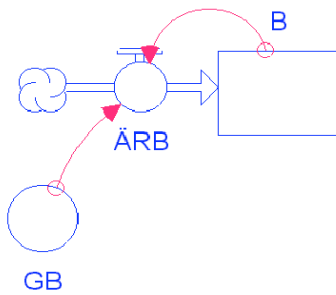
### Powersim



Powersim ist für Simulationen größerer Zeiträume ausgelegt. Das macht sich sehr unangenehm in der Beschriftung der Zeitachse bemerkbar. Für die Beschriftung der Zeitachse wird das volle Datum benutzt. Ich konnte bisher keinen anderen Beschriftungsmodus einsetzen.

Alle Größen in Powersim haben Einheiten. Eine einheitenlose Simulation lässt sich in Powersim nicht durchführen. Beim Einsatz in der Schule dürfte das zu unnötigen Schwierigkeiten führen. Die Benutzerführung von Powersim ist in englischer Sprache. Gut ist die grafische Darstellung sowohl der Modelle als auch der Schaubilder.

**Stella**



Auch Stella ist, wie Powersim, ein Profitool mit englischer Benutzerführung. Im Vergleich zu Powersim rechnet Stella mit und ohne Einheiten.

Bestandsgrößen und Änderungsraten können in Stelle standardmäßig nur positive Werte annehmen. Im Normalfall müssen hier vom Benutzer explizit negative Werte zugelassen werden. Das wird in vielen Fällen zu Verwirrung führen.

**Moebius**

Moebius ist sehr leicht bedienbar. Die Modelle lassen sich nicht nur mit den üblichen grafischen Elementen programmieren, sondern auch in einer symbolischen, sowie in einer alltagsnahen Sprache. Moebius beschränkt sich bewusst auf das Eulerverfahren.

Das Programm ist erhältlich über <http://www.mintext.de/>.

A screenshot of the Moebius software interface. The main window shows a graph with a y-axis labeled 'S' and 'DS' and an x-axis labeled 'ZEIT'. A blue curve starts at approximately (0, 80) and decreases towards the x-axis. An 'Algorithmische Modellierung' dialog box is open, showing the following code:
 

```

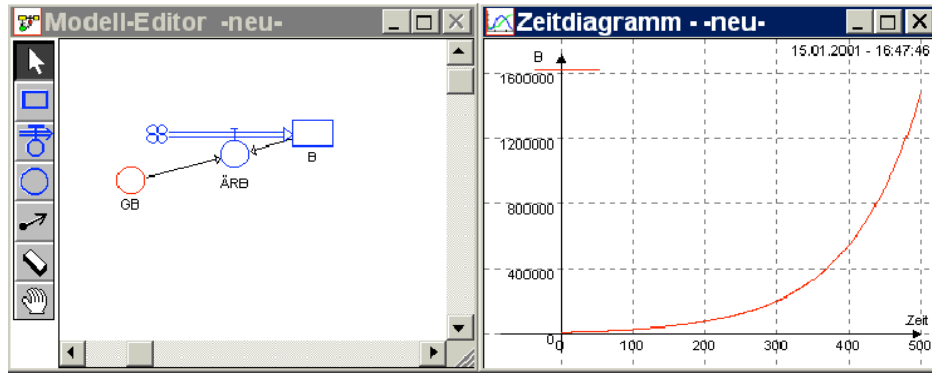
Wiederhole 100 mal
ds:=k*s
s:=s+ds
ZEIT:=ZEIT+1
    
```

 The dialog box also includes a 'Startwerte' table:
 

K	-0,1
S	100
ZEIT	0

 The interface includes a menu bar, a toolbar, and a status bar at the bottom that reads 'Unregistrierte Shareware-Version: Verwendung im Unterricht nicht erlaubt!'.

**Dynasys**

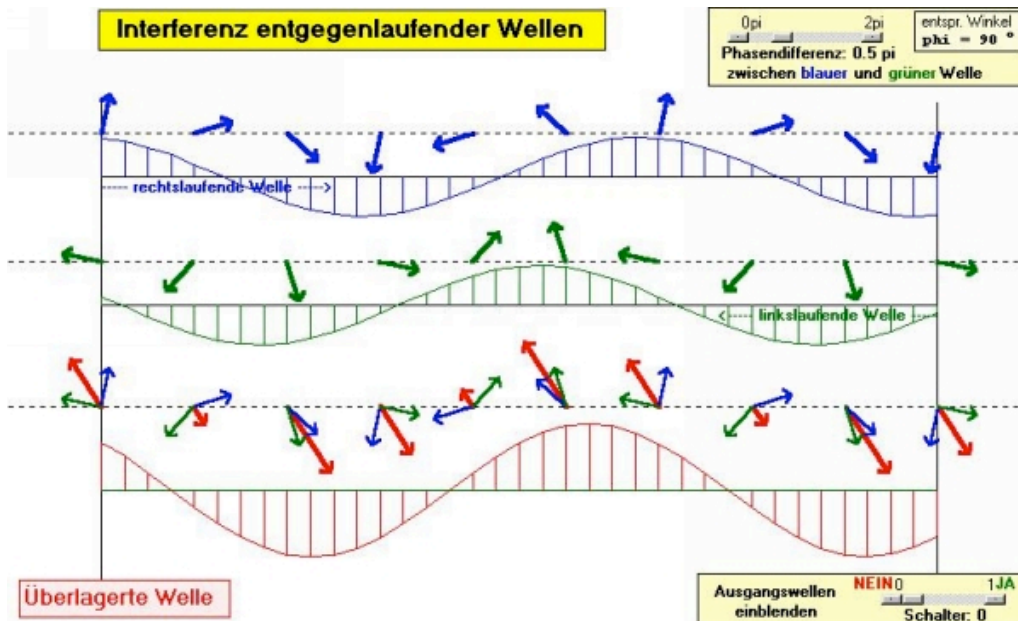


Die Modelle dieses Artikels wurden mit Dynasys erstellt. Auch Dynasys hat wie Modus eine deutsche Benutzerführung. Es nutzt die Rechnerressourcen sehr gut aus. So lassen sich auch komplexe Modelle auf eher langsamen Rechnern praktisch ohne Zeitverlust simulieren.

Die grafische Darstellung der Modelle lässt sich weder vergrößern noch verkleinern, was bei hohen Bildschirmauflösungen unangenehm ist.

**PAKMA**

Die Software PAKMA liegt dem Lösungsbuch zu Dorn-Bader Physik 12/13 bei. Der bisher von mir am häufigsten genutzte Einsatzbereich von PAKMA ist das Zeigen und Analysieren von fertig erstellten Simulationen. Hier werden physikalische Größen und relevante Zusammenhänge durch dynamisch ikonische Repräsentationen visualisiert.



In einem weiteren Einsatzbereich können Lernende mit PAKMA auch physikalische Modelle erstellen. Hierbei kann man beliebige physikalische Sachverhalte numerisch iterativ berechnen lassen und graphisch darstellen.

Diese Modellbildung ist nicht nur gleichungsorientiert möglich. Seit kurzem besteht mit dem Zusatzmodul VisEdit auch die Möglichkeit, Wirkungsgefüge am Bildschirm zu entwerfen und dies automatisch in ein lauffähiges Kernprogramm umzusetzen. Gegenüber anderen Modellbildungssystemen bietet PAKMA mit dieser Erweiterung zwei entscheidende Vorteile: Einerseits können auch Messvorgänge und ihre Auswertung mit in zu erstellende Netzwerkgefüge integriert werden, so dass Messwerterfassung mit Auswertung und Modellbildung parallel ablaufen und ihre Ergebnisse in Realzeit verglichen werden können, so dass sich sofort Aussagen über die Angemessenheit des zugrunde gelegten Modells ergeben. Andererseits können die Ergebnisse eines ablaufenden Modells nicht nur durch Graphen sondern auch durch dynamisch ikonische Repräsentationen visualisiert werden.

## Numerische Verfahren

Grundsätzlich kann man sagen, dass das **Euler-Verfahren** zur Einführung in die Modellbildung am besten geeignet ist, da es direkt aus der Anschauung folgt. Leider ist es auch das ungenaueste Verfahren. Es nähert die simulierte Funktion lediglich linear an.

### Verfahren höherer Ordnung

Beim **verbesserten Euler-Verfahren** geht man davon aus, dass der Mittelwert

$$\frac{1}{2} \cdot (f(t_k, y_k) + f(t_{k+1}, y_{k+1}))$$

in der Regel eine bessere Näherung für die Steigung  $\frac{1}{h} \cdot (y(t_{k+1}) - y(t_k))$

darstellt. Den zunächst noch unbekanntem Wert  $y_{k+1}$  im zweiten Summanden approximiert man mit dem Euler-Verfahren:

$$F(t_k, y_k, h) = \frac{1}{2} \cdot (K_1 + K_2) \quad \text{mit } K_1 = f(t_k, y_k), K_2 = f(t_k + h, y_k + h \cdot K_1)$$

Die simulierte Funktion wird als Taylor-Entwicklung der Ordnung 2 angenähert.

Das **Verfahren von Runge und Kutta** verwendet zur Berechnung der Ableitung F eine weitere Stützstelle  $t_k + 0,5 \cdot h$  und ein gewichtetes Mittel aus 4 Funktionswerten von f:

$$F(t_k, y_k, h) = \frac{1}{6} \cdot (K_1 + 2 \cdot K_2 + 2 \cdot K_3 + K_4)$$

$$\text{mit } K_1 = f(t_k, y_k), \quad K_2 = f(t_k + \frac{1}{2} \cdot h, y_k + \frac{1}{2} \cdot h \cdot K_1),$$

$$K_3 = f(t_k + \frac{1}{2} \cdot h, y_k + \frac{1}{2} \cdot h \cdot K_2), \quad K_4 = f(t_k + h, y_k + h \cdot K_3)$$

Es nähert die simulierte Funktion als Taylor-Entwicklung der Ordnung 4 an.

### Fehlerabschätzungen

Die letzten beiden Verfahren erfordern gegenüber dem Euler-Verfahren einen doppelten bzw. vierfachen Rechenaufwand. Sie liefern aber, falls f genügend oft differenzierbar ist, bei gleicher

Schrittweite um Größenordnungen genauere Ergebnisse, da die Steigung  $\frac{1}{h} \cdot (f(t_k + h) - f(t_k))$

durch  $F(t_k, y_k, h)$  höherer Ordnung approximiert wird.

Um dies quantitativ zu erfassen, entwickelt man die exakte Lösung  $y(t + h)$  und  $F(t, y(t), h)$  um t in Taylor-Reihen nach h. Es ergibt sich die folgende Entwicklung nach Potenzen:

$$y(t + h) = y(t) + h \cdot f(t, y(t)) + \frac{h^2}{2} \cdot (f_t + f_y \cdot f)|_{(t, y(t))} + \dots$$

Man vergleicht dies mit der h-Reihe von  $F(t, y(t), h)$  und sagt, das Verfahren besitzt die (**Konsistenz-)**Ordnung p, wenn es Konstanten  $h_0, C > 0$  gibt, so dass

$$\left| \frac{y(t + h) - y(t)}{h} - F(t, y(t), h) \right| \leq C \cdot h^p \quad \text{für } |h| < h_0.$$

Danach hat das Euler-Verfahren die Ordnung 1, das verbesserte Euler-Verfahren die Ordnung 2 und das Verfahren von Runge und Kutta sogar die Ordnung 4.

In der Praxis interessiert stets der globale Fehler  $|y(t) - y_n|$ , der nach n Schritten an der Stelle  $t = t_n$  auftritt. Dieser setzt sich aus 2 Anteilen zusammen:

**Verfahrens- oder Diskretisierungsfehler**, die sich durch Häufung der lokalen Fehler ergeben, indem die exakte Lösung  $y(t)$  durch die Diskretisierung  $y_n$  ersetzt wird.

Ist f p-mal stetig partiell differenzierbar und hat das Verfahren die Ordnung p mit stetiger Funktion  $F(t, y, h)$ , die in y einer L-Bedingung

$$|F(t, y_1, h) - F(t, y_2, h)| \leq L |y_1 - y_2|, \quad \text{für ein } L > 0 \text{ genügt, so gilt } |y_n - y(t_n)| \leq C \cdot h^p \cdot \frac{e^{n \cdot L \cdot h} - 1}{L}.$$

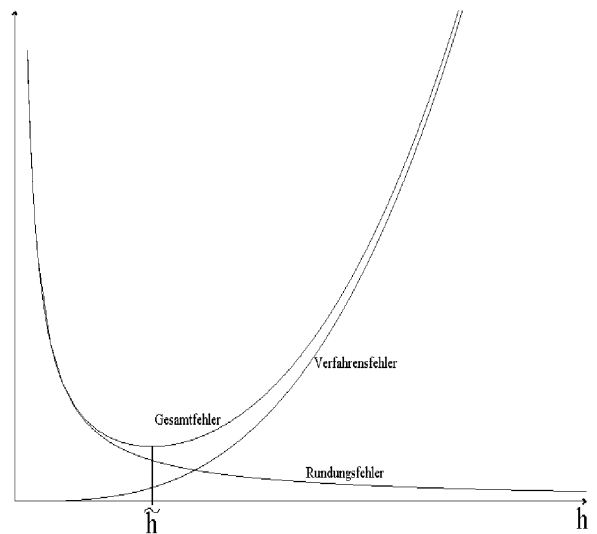
Wird die Schrittweite  $h$  halbiert, so wird auch der Verfahrensfehler beim Euler-Verfahren halbiert, beim verbesserten Euler-Verfahren geviertelt und beim Runge-Kutta-Verfahren geht er sogar auf  $\frac{1}{16}$  zurück.

**Rundungsfehler**, die in der Zahlendarstellung von Digitalrechnern mit endlicher Genauigkeit unvermeidbar sind. Bezeichnet  $y_n^*$  den tatsächlich errechneten Wert, so gilt für unsere Verfahren unter den vorhergenannten Voraussetzungen für  $f$  und  $F$ :

$$|y_n - y_n^*| \leq \frac{D}{h} \cdot \frac{e^{n \cdot L \cdot h} - 1}{L} \text{ mit einem } D > 0.$$

Der **Gesamtfehler**  $|y_n - y_n^*| + |y_n - y(t_n)|$  hat abhängig von  $h > 0$  bei festem  $t_n \leq t_0 = n \cdot h$  den skizzierten qualitativen Verlauf. Daraus lassen sich 2 grundsätzliche Erkenntnisse ablesen:

1. Der Gesamtfehler kann nicht beliebig klein gemacht werden. Es gibt eine optimale Schrittweite  $\tilde{h}$  mit minimalem Gesamtfehler.
2. Ein Verfahren höherer Ordnung führt mit weniger Schritten (größeres  $\tilde{h}$ ) zu einem kleineren minimalen Gesamtfehler. Der kleinere Verfahrensfehler wiegt den höheren Rechenaufwand pro Schritt bei weitem auf.



In der rechten Skizze ist die numerische Berechnung von  $y(1,8)$  für das Anfangswertproblem

$$y' = 1 + (y - t)^2, y(0) = 0,5$$

mit der Lösung  $y(t) = t + \frac{1}{2 \sqrt{x}}$  dargestellt. Man

sieht deutlich die Überlegenheit des Verfahrens von Runge und Kutta gegenüber dem Euler-Verfahren.

