

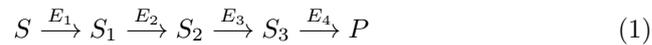
---

Von der Mathematischen Biologie zur Systembiologie  
(Vorlesung Prof. Dr. J. Timmer)  
Aufgabenzettel Nr. 8

---

**Aufgabe 12 (Übung): Kette von Enzymreaktionen**

Betrachten Sie die folgende Kette von Michaelis-Menten Enzymreaktionen:



Implementieren Sie das zugehörige dynamische System. Dabei sei  $S = 1$  eine konstante Konzentration und für die Enzyme gilt:

$$\begin{aligned} E_1 : v_{\max}^1 &= 0,1; & K_M^1 &= 0,1 \\ E_2 : v_{\max}^2 &= 1,0; & K_M^2 &= 1,0 \\ E_3 : v_{\max}^3 &= 1,0; & K_M^3 &= 0,1 \\ E_4 : v_{\max}^4 &= 5,0; & K_M^4 &= 5,0 \end{aligned}$$

- Bestimmen Sie die Gleichgewichtskonzentrationen  $S_1, S_2$  und  $S_3$ , sowie den Gleichgewichtsfluß  $J$ , indem Sie das System für hinreichend lange Zeiten simulieren.
- Veranschaulichen Sie die Gleichgewichtskonzentrationen auf Grundlage der gegebenen Parameter.
- Bestimmen Sie die Konzentrations- und Fluß-Kontrollkoeffizienten.

*Tips:*

- Integrieren Sie das System und merken Sie sich die Gleichgewichtswerte der Flüsse und Konzentrationen.
  - Ändern Sie die einzelnen Reaktionsgeschwindigkeiten in der Integration um jeweils 1%, und wählen Sie die Gleichgewichtskonzentration als Startwerte der Integration.
  - Bestimmen Sie die Kontrollkoeffizienten durch finite Differenzen.
- Visualisieren Sie das Ergebnis.

**Tips und Tricks zu Graphiken mit R**

Das Erstellen von ansprechenden Graphiken erfordert im allgemeinen mehr als das bloße anwenden des `matplot()` Befehls. Im wesentlichen gib es drei Punkte, die man beachten sollte:

- i.) Die Darstellung der Daten(punkte): Die wichtigen Optionen, die `matplot()` hat, sind `type`, `lty`, `lwd`, `lend`, `pch`, `col`. Eine deutliche Erweiterung dessen bietet der Befehl `par()`, den man bei Bedarf vor dem Plotbefehl ausführen kann. Desweiteren ist es manchmal wünschenswert die Daten in einem logarithmischen Koordinatensystem darzustellen. Dies geschieht mit der Option `log = "x"` oder `log = "y"` für die Abzisse oder die Ordinate.

- ii.) Die Beschriftung der Graphen: Dazu stehen `matplot()` einige Optionen zur Verfügung. Die wichtigsten sind `xlab`, `ylab`, `main`. Desweiteren kann man mit dem Befehl `legend()` eine Legende zu einer Graphik erstellen. Nützlich ist dabei der Befehl `expression()` mit dem man nahezu beliebige mathematische Formeln / Zeichen darstellen kann. Eine Demonstration dessen kann man sich mit `demo(plotmath)` ansehen.
- iii.) Die Farbwahl. Für einfache Graphen ist es oft (fast) egal in welcher Farbe man diese darstellt. Nichtsdestotrotz sollte man einige Punkte beachten:
- Kann man Farben in einem s/w-Ausdruck auseinanderhalten?
  - Sind die Farben vortragsgeeignet, d.h. kann ein Beamer die Farben darstellen und können ggf. Menschen mit Farbenfehlsichtigkeit (Farbenblindheit), immerhin 5% der Bevölkerung(!), die Farben auseinanderhalten?
  - Suggestieren die benutzten Farben das Richtige? Dies ist vor allem beim Einfärben von Flächen (z.B. Konturplots) von Bedeutung.
  - Sind die Farben angenehm für das Auge? Dies ist insbesondere bei voll saturiert dargestellten Flächen nicht der Fall.

Es gibt z.B. das R-Paket *colorspace*, das Farbpaletten im HCL-Raum (hue, chroma, luminance) zur Verfügung stellt. Der Farbraum ist geeignet um Daten aller Art darzustellen. Mehr Informationen dazu gibt es hier:  
<http://cran.r-project.org/web/packages/colorspace/index.html>.

Desweiteren kann man mit R natürlich nicht nur einfache Graphen darstellen. Oft gebrauchte und nützliche Funktionen sind: `hist()` für Histogramme, `image()` und `filled.contour()` für Konturplots, `barplot()` für Balkendiagramme, `boxplot()` für Boxplots, `pie()` für Tortendiagramme, `stripchart()` für Liniendiagramme, `dotchart()` für Punktdiagramme uvm. Für alle, die damit noch nicht zufrieden sind gibt es noch das Paket *lattice*, das noch mehr Optionen offen hält...

## Das Speichern von Graphiken

Ein einfacher (aber wenig flexibler) Weg Graphiken zu speichern ist der Befehl `savePlot()`. Besser ist es Graphiken in ein *device* zu drucken. Dies geschieht zum Beispiel so:

```
> pdf("filename.pdf", height = 5, width = 5)
> matplot(...)
> lines(...)
> legend(...)
> dev.off()
```

Dies druckt alles bis zum Befehl `dev.off()` in die Datei `filename.pdf` statt auf den Bildschirm. Man kann auf diese Weise auch mehr als eine Graphik in ein *device* drucken. Statt als pdf mit `pdf()` kann man auch in andere Formate z.B. png mit `png()` drucken.