Anwendung von (Mathematica und) Matlab in der Physikalischen Chemie

11. Präsentation möglicher Lösungen für die gestellten Aufgaben

UNI FREIBURG

Albert-Ludwigs-Universität Freiburg

Dr. Till Biskup

Institut für Physikalische Chemie
Albert-Ludwigs-Universität Freiburg
Wintersemester 2017/2018

Ein reales Beispiel: Fluoreszenz-Versuch aus dem PCG

- Ausgangslage
  - Daten wurden alle gemessen
  - Daten liegen als Textdateien (ASCII) vor
- Zielstellung
  - Vollständige Auswertung gemäß Fragestellung
  - Abbildungen, die den Assistenten zufriedenstellen (und den wissenschaftlichen Standards entsprechen)
- Vorgehen
  1. Pflichtenheft erstellen (was muss getan werden?) ✔
  2. Notwendige Grundlagen von Matlab aneignen ✔
  3. Auswertung gemäß Pflichtenheft programmieren ✔

Pflichtenheft: Einzelne Schritte der Auswertung

- Daten einlesen
  - Daten importieren in Matlab
- Spektren darstellen
  - Daten in Matlab grafisch darstellen (plotten)
  - Achsenbeschriftungen gemäß Vorgaben
  - Abbildungen aus Matlab exportieren
- Intensitäten für eine Wellenlänge aus mehreren Spektren
  - Mehrere Spektren einlesen
  - Zugriff auf einen bestimmten Wert in einem Vektor
- Lineare und nichtlineare Kurvenanpassung
  - Matlab-Routinen zur Kurvenanpassung

Schritte in die Praxis

Ein paar Worte zu den im Folgenden vorgestellten Lösungen

UNI FREIBURG

- Viele Wege führen nach Rom.
  - Es gibt viele verschiedene Möglichkeiten, die gestellten Aufgaben zu lösen.
  - Die nachfolgend vorgestellten Lösungen sind nicht unbedingt die besten.
- Hilfe zur Selbsthilfe
  - Programmieren lässt sich nur durch praktische Arbeit erlernen.
  - Lösungen im Detail (in *jedem* Schritt) zu verstehen, ist zentral für eigene Fortschritte.
  - Aufgaben und Lösungen sollen zeigen, dass Matlab bei realen Problemen helfen kann.

Übersicht

UNI FREIBURG

Daten importieren

Anregungsspektrum darstellen

Streubande identifizieren

Intensität als Funktion der Konzentration

Kurvenanpassungen

Abbildungen exportieren

Wie sehen die Dateien aus?

- Dezimaltrennzeichen
  - Punkt
- Anzahl der Zeilen im Dateikopf
  - 54
- Spaltentrennzeichen
  - Tabulator

☛ Wir können die Funktion `importdata` verwenden.

Daten importieren

Ein Einzeiler – dank `importdata`

UNI FREIBURG

Listing 1: Befehl zum Datenimport

```
1 data = importdata('1A-01.sp','\t',54);
```

Listing 2: Matlab-Rückgabe auf der Kommandozeile

```
1 >> data = importdata('1A-01.sp','\t',54)
2
3 data =
4
5          data: [261x2 double]
6     textdata: {54x1 cell}
7
8 >>
```

Daten importieren

Die Funktion `LS45ASCIIread.m`

UNI FREIBURG

Listing 3: LS45ASCIIread.m

```
1  function data = LS45ASCIIread(filename)
2  % LS45ASCIIREAD Load ASCII export from PerkinElmer LS45 spectrometer.
3  %
4  % Usage:
5  %   data = LS45ASCIIread(filename)
6  %
7  %   filename - string
8  %              Name of file to read
9  %
10 %   data     - struct
11 %              data     - numerical data
12 %              textdata - header information
13
14 % Copyright (c) 2016, Till Biskup <till.biskup@physchem.uni-freiburg.de>
15 % 2016-02-10
16
17 separator = '\t';
18 nheaderlines = 54;
19
20 data = importdata(filename,separator,nheaderlines);
21
22 end
```

Warum eine eigene Funktion für einen effektiven Einzeiler?

Listing 4: Vergleich von Funktionsaufruf und direktem Import

```
data = LS45ASCIIread('1A-01.sp');
data = importdata('1A-01.sp','\t',54);
```

- Weil es Code leserlicher macht.
  - In beiden Fällen werden offensichtlich Daten importiert, aber `LS45ASCIIread` verrät, welche Art von Daten.
- Weil man nur einmal denken muss.
  - Spaltentrenner und Anzahl der Kommentarzeilen müssen nur einmal in der Funktion definiert werden.
  - Der Nutzer der Funktion muss sich nicht darum kümmern.

Anregungsspektrum darstellen

Ein einfacher `plot`-Befehl und ein bisschen Kosmetik

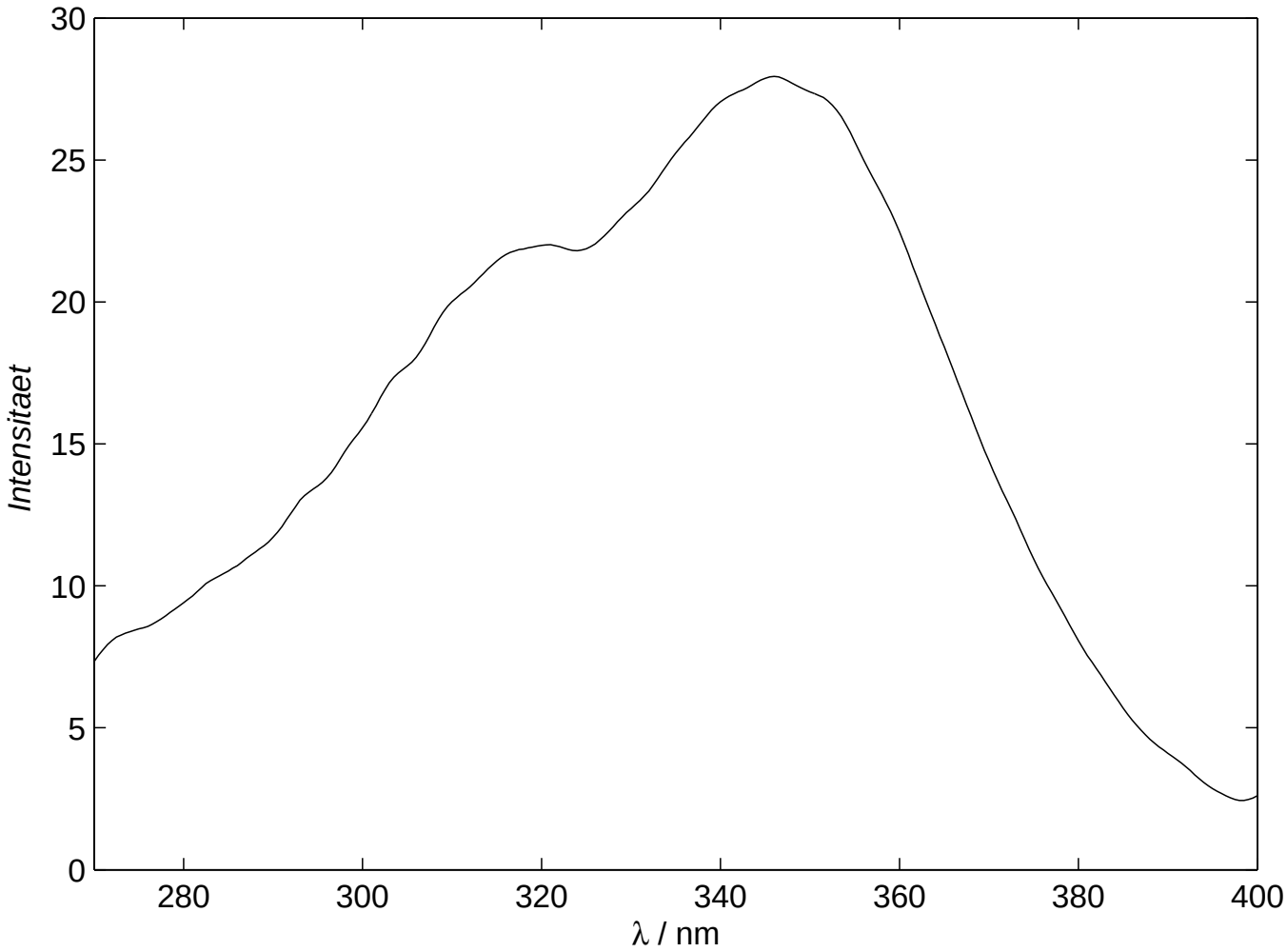
UNI FREIBURG

Listing 5: Anregungsspektrum darstellen

```
1  % Daten importieren
2  data = LS45ASCIIread('1A-01.sp');
3
4  % Einfacher Plot
5  % Die Daten liegen in data.data in zwei Spalten vor
6  plot(data.data(:,1),data.data(:,2),'k');
7
8  % Achsenbeschriftungen
9  xlabel('{\it\lambda} / nm');
10 ylabel('{\it Intensitaet}');
11
12 % ZUSATZ: Dimension der x-Achse anpassen
13 set(gca,'XLim',[min(data.data(:,1)),max(data.data(:,1))]);
```

Anregungsspektrum darstellen

Das Ergebnis erster Plotversuche



Anregungsspektrum darstellen

Hervorheben des Intensitätsmaximums

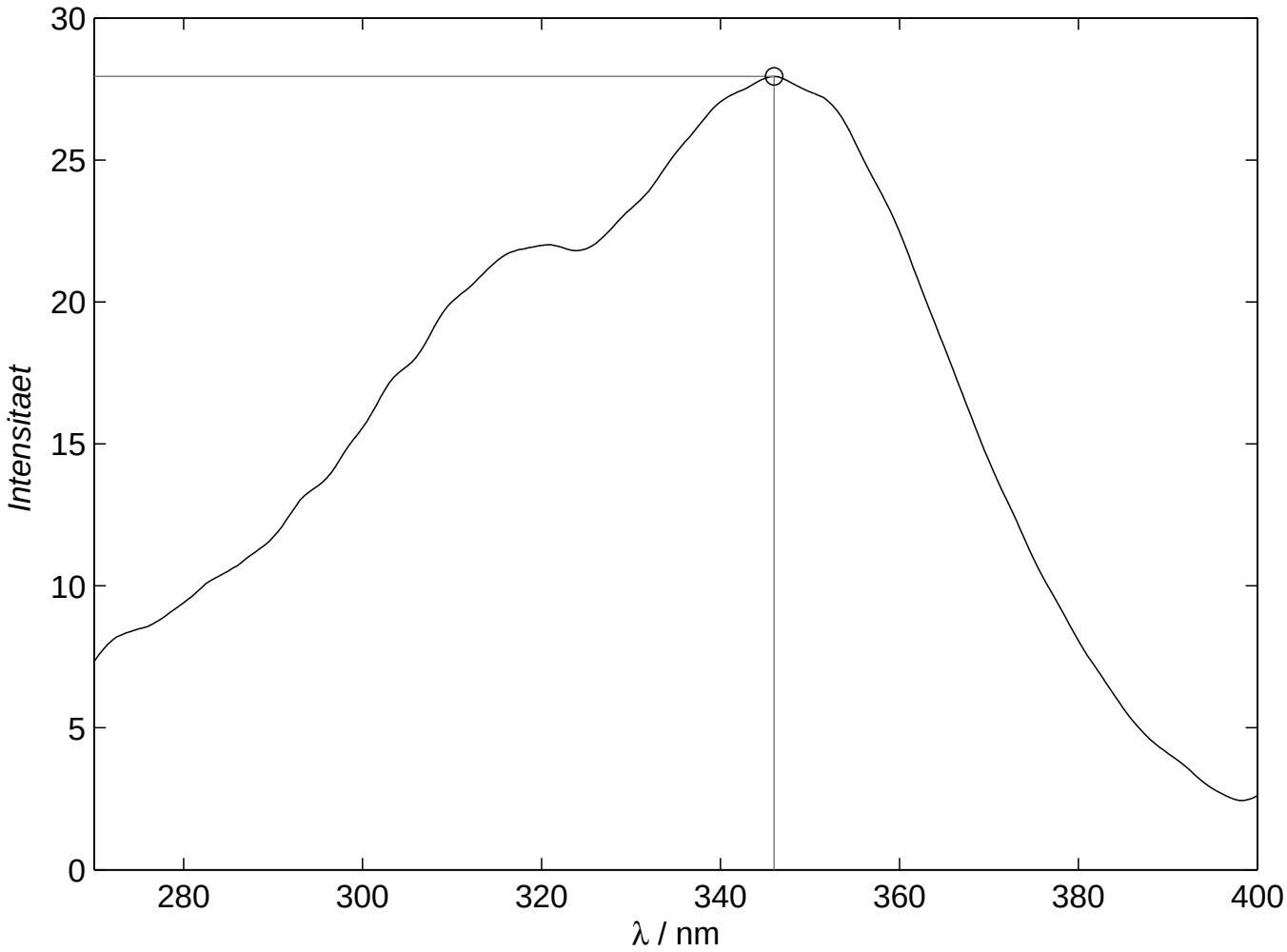
UNI FREIBURG

Listing 6: Intensitätsmaximum des Anregungsspektrums hervorheben

```
% Wert und Index des Maximums
[maxValue,maxIndex] = max(data.data(:,2));

% Wellenlaenge des Maximums
lambdaMax = data.data(maxIndex,1);

% Senkrechte Linie einzeichnen
line([lambdaMax,lambdaMax],[0,maxValue],'color',[.5 .5 .5]);

% Horizontale Linie einzeichnen
line([lambdaMax,data.data(1,1)],[maxValue,maxValue],'color',[.5 .5 .5]);

% ZUSATZ: Anregungsmaximum mit Kreis hervorheben
hold on;
plot(lambdaMax,maxValue,'ko');
hold off;
```

Anregungsspektrum darstellen

Das Ergebnis in grafischer Form



Streubande identifizieren

Mehrere Kurven mit einem `plot`-Befehl darstellen, Legende

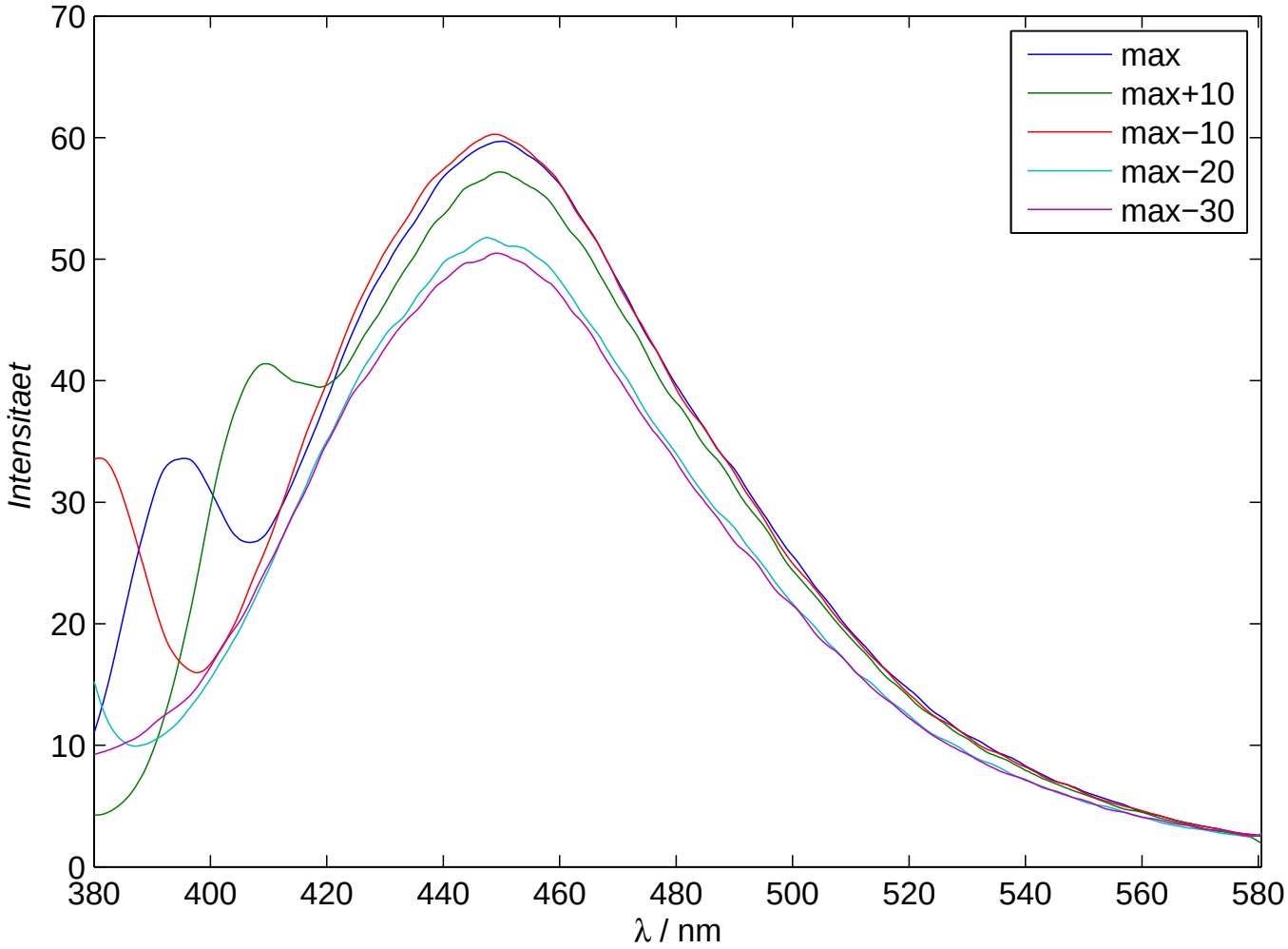
UNI FREIBURG

Listing 7: Anregungsspektrum darstellen

```
% Daten importieren
spek1 = LS45ASCIIread('1E-01.sp');
spek2 = LS45ASCIIread('1E-02.sp');
spek3 = LS45ASCIIread('1E-03.sp');
spek4 = LS45ASCIIread('1E-04.sp');
spek5 = LS45ASCIIread('1E-05.sp');

% Einfacher Plot
plot(...
    spek1.data(:,1),spek1.data(:,2),...
    spek2.data(:,1),spek2.data(:,2),...
    spek3.data(:,1),spek3.data(:,2),...
    spek4.data(:,1),spek4.data(:,2),...
    spek5.data(:,1),spek5.data(:,2));

% Achsenbeschriftungen
xlabel('{\it\lambda} / nm');
ylabel('{\it Intensitaet}');

% Dimension der x-Achse anpassen
set(gca,'XLim',[min(spek1.data(:,1)),max(spek1.data(:,1))]);

% Legende
legend({'max','max+10','max-10','max-20','max-30'},'Location','ne');
```

Streubande identifizieren

Das Ergebnis in grafischer Form



Intensität als Funktion der Konzentration

Daten laden und Intensitäten extrahieren

UNI FREIBURG

Listing 8: Daten laden und Intensitäten extrahieren

```
1  dateinamen = {'2E-01.sp','2E-02.sp','2E-03.sp','2E-04.sp','2E-05.sp',...
2      '2E-06.sp','2E-07.sp','2E-08.sp','2E-09.sp','2E-10.sp','2E-11.sp'};
3
4  for messung = 1:length(dateinamen)
5      data = LS45ASCIIread(dateinamen{messung});
6      % Intensitaet bei 448 nm
7      Imax(messung) = data.data(data.data(:,1)==448,2);
8  end
9
10 % Korrektur der Intensitaeten mit Blindprobe
11 Imax = Imax-Imax(1);
```

Intensität als Funktion der Konzentration

Konzentration berechnen

UNI FREIBURG

Listing 9: Konzentration berechnen

```
1  % Volumina der beiden Loesungen
2  VChinin = [ 290 600 : 300 : 1500 2000 3000 40 50 60 ];
3  VPuffer = [ 3210 2900 : -300 : 2000 1500 500 3000 3000 3000 ];
4
5  % Konzentration von Chinin in ppm
6  cChinin = VChinin./(VPuffer+VChinin);
7
8  % Korrektur um den Faktor 100 fuer die letzten drei Punkte
9  cChinin(end-2:end) = cChinin(end-2:end)*100;
10
11 % Molekulargewicht von Chinin (freie Chininbase) in g/mol
12 MwChinin = 324.41;
13
14 % Konvertierung ppm -> mol/L
15 cChinin = cChinin*1e-4/MwChinin;
```

Intensität als Funktion der Konzentration

Daten auftragen, Punkte dabei nicht verbinden

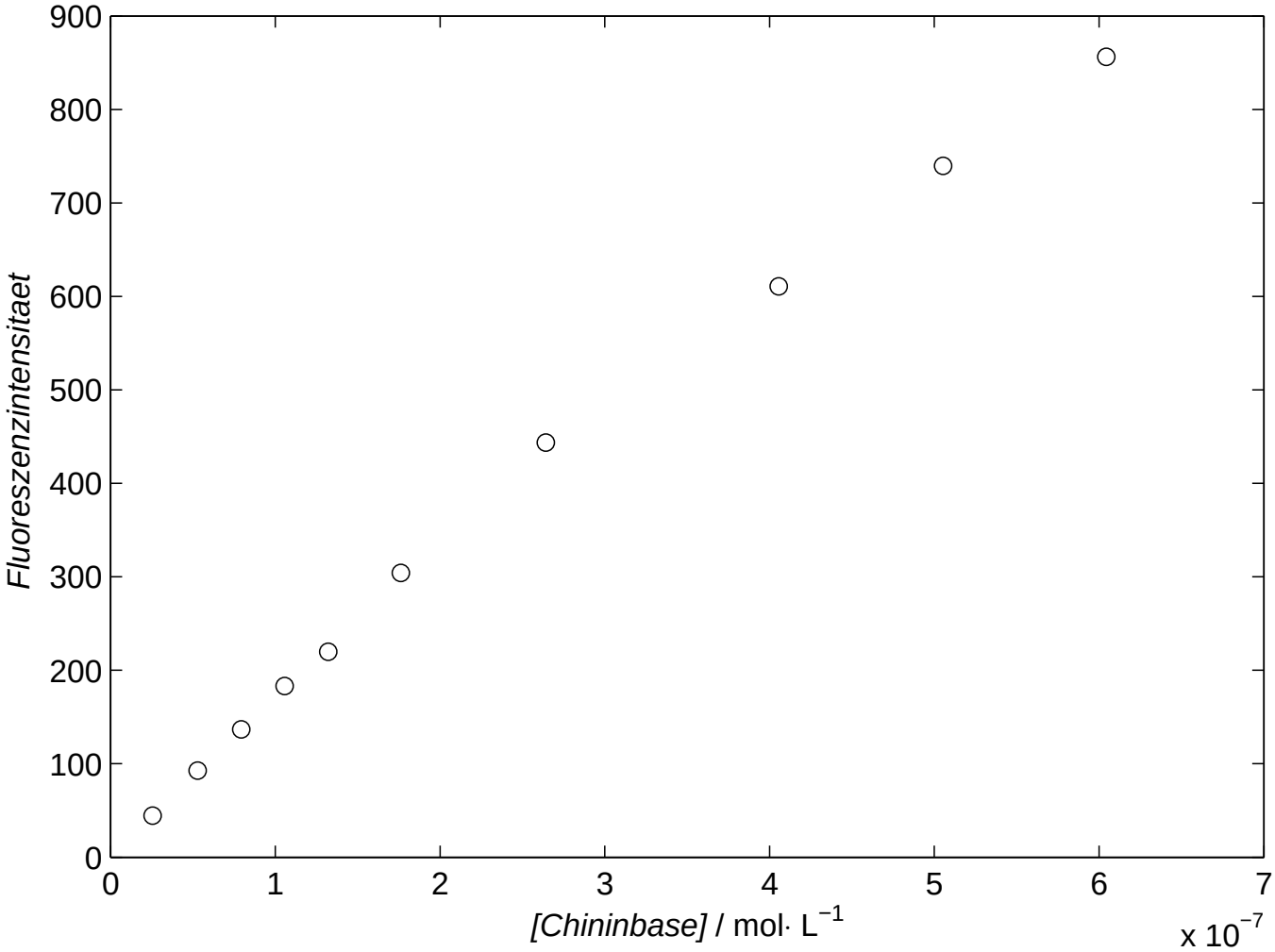
UNI FREIBURG

Listing 10: Darstellung der Konzentrationsabhängigkeit

```
1 % Punkte nicht verbinden, auf die Laenge der Vektoren achten
2 plot(cChinin,Imax(2:end),'ko');
3
4 % Achsenbeschriftungen
5 xlabel('{\it[Chininbase]} / mol\cdot L^{-1}');
6 ylabel('{\it Fluoreszenzintensitaet}');
```

Intensität als Funktion der Konzentration

Das Ergebnis in grafischer Form



Kurvenanpassungen

Lineare Regression durch die ersten vier Datenpunkte

UNI FREIBURG

Listing 11: Lineare Regression durch die ersten vier Datenpunkte

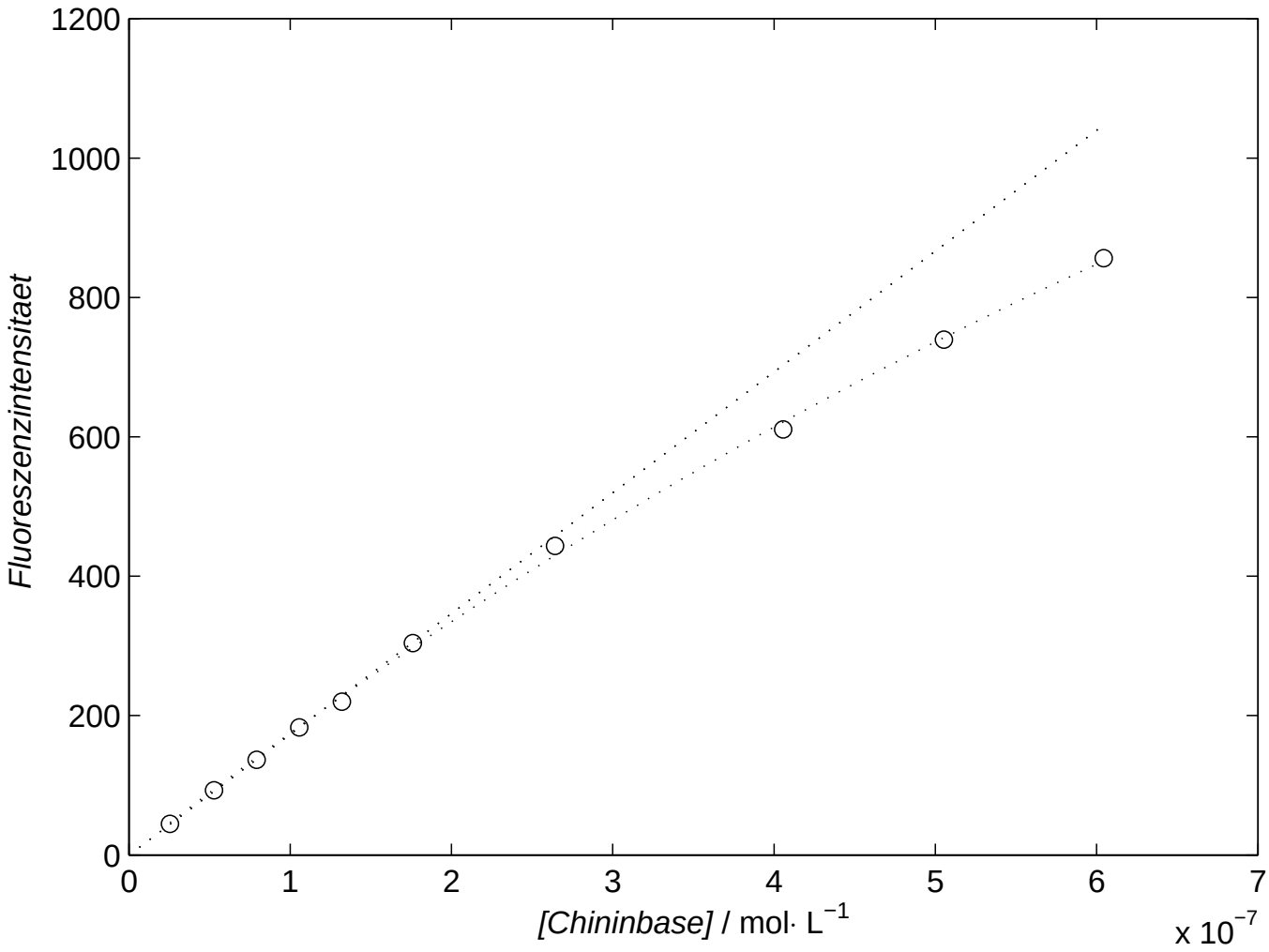
```
1  % Lineare Regression durch die ersten vier Punkte
2  [anstieg,fehler] = lscov(cChinin(1:4)',Imax(2:5)');
3
4  % ALTERNATIVE: Backslash-Operator
5  anstieg = cChinin(1:4)' \ Imax(2:5)';
6
7  % Berechnung der Werte der linearen Regression: y = anstieg * x
8  x = [0 cChinin(end)];
9  y = anstieg * x;
10
11 % Darstellung der linearen Regression
12 hold on;
13 plot(x,y,'k:');
14 hold off;
```

Listing 12: Nichtlineare Kurvenanpassung

```
1  % Funktion, die angepasst werden soll: f(x) = a * (1-10^(-b*x))
2  fun = @(a,x)a(1).*(1-10.^(-a(2).*x));
3
4  % Minimierungsfunktion: Summe der Quadrate der Abweichungen
5  fitfun = @(a)sum((Imax(2:end)-fun(a,cChinin)).^2);
6
7  % Startwerte defininieren
8  a(1) = 1e3;
9  a(2) = 5e4;
10
11 % Minimierung
12 fita = fminsearch(fitfun,a);
13
14 % X-Werte und Funktion fuer die glatte Darstellung der Kurve
15 xWerte = cChinin(1):(cChinin(end)-cChinin(1))/100:cChinin(end);
16
17 % Darstellung der nichtlinearen Kurvenanpassung
18 hold on;
19 plot(xWerte,fun(fita,xWerte),'k:');
20 hold off;
```

Kurvenanpassungen

Das Ergebnis in grafischer Form



Abbildungen exportieren

Ein paar grundsätzliche Überlegungen

UNI FREIBURG

Warum Abbildungen automatisiert exportieren?

- Sorgt für ein möglichst konsistentes Aussehen.
- Erleichtert den Reexport nach Änderungen an den Daten.

Warum Abbildungen als PDF-Dateien exportieren?

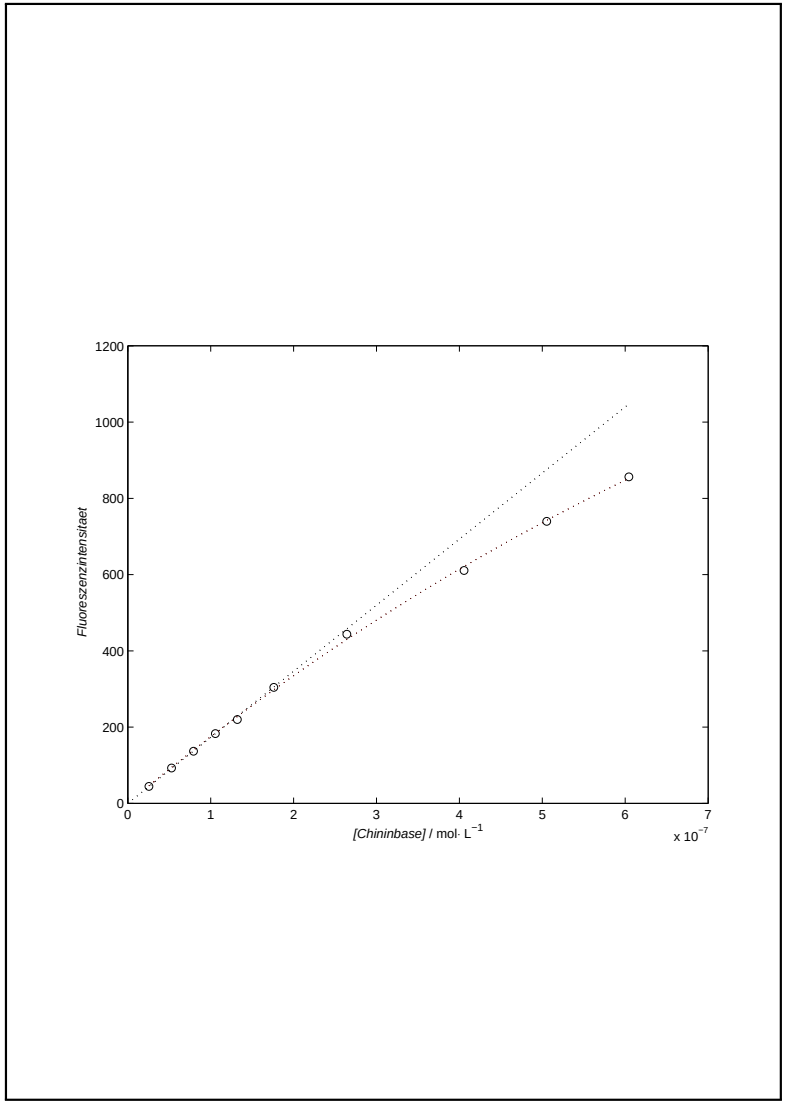
- PDF-Dateien sind (in der Regel) vektorisiert, können also beliebig skaliert und einfach nachbearbeitet werden.

Listing 13: Grundlegender Abbildungsexport als PDF-Datei

```
1 print(gcf,'erster-test.pdf','-dpdf');
```

Abbildungen exportieren

Der erste Versuch – noch nicht ganz das gewünschte Ergebnis



Abbildungen exportieren

Grundlegendes zu Abbildungen in Matlab

UNI FREIBURG

- Abbildungen sind (so etwas wie) Objekte
  - Objekte haben Eigenschaften (*properties*).
  - Eigenschaften können abgefragt oder gesetzt werden.
  - Objekte haben eine Referenz (*handle*) für den Zugriff.
- Funktionen zum Arbeiten mit Eigenschaften
  - `get` fragt eine Eigenschaft ab
  - `set` setzt eine Eigenschaft
- Spezielle Referenzen für Grafikobjekte
  - `gcf` aktives Abbildungsfenster (*current figure*)
  - `gca` aktive Achse (*current axes*)
  - `gco` aktives Grafikobjekt (*current object*)

Abbildungen exportieren

Auf dem Weg zu sinnvollem Abbildungsexport

UNI FREIBURG

Listing 14: Anpassungen der Seitengröße

```
1  % Anpassung der Seitengroesse
2  set(gcf,'paperunits','centimeters');
3  set(gcf,'papersize',[16 10]);
4
5  % Anpassung der Positionierung auf der Seite
6  set(gcf,'paperpositionmode','auto');
7  set(gcf,'Units','centimeters');
8
9  % Anpassung der Groesse der Achsen
10 set(gca,'Units','centimeters');
11 set(gca,'OuterPosition',[0 0 16 10]);
12
13 % Positionierung auf dem Papier
14 oldpos = get(gcf,'Position');
15 set(gcf,'Position',[oldpos([1 2]) 16 10]);
```

- Die Reihenfolge der Befehle ist nicht immer egal.
- Manchmal erschließt sich die Logik nicht zwangsläufig...

Ausblick

Wie geht's weiter?

UNI FREIBURG

...gleich geht's weiter

Vorschau: Matlab im echten Leben

- Automatisierung von Routineaufgaben
- Interaktive Datenauswertung (Kommandozeile)
- Interaktive Datenauswertung (grafisch)
- Ausblick auf weiterführende Veranstaltungen