QUICK TUTORIALS

SPECTR-O-MATIC TOOLBOX

TABLE OF CONTENTS

Create specdata objects

Operate with spectra

Select and search

Organize data by categories

CREATE SPECDATA OBJECTS

SPECTR-O-MATIC TOOLBOX

CREATE SPECDATA FROM VARIABLES

1. Create some X and Y arrays:

```
x = 0:0.1:pi;
y = sin(x);
```

x *1x32 double*

0	0.1	0.2	0.3	0.4	…	3	3.1

y *1x32 double*

0	0.1	0.199	0.296	0.389	…	0.141	0.042

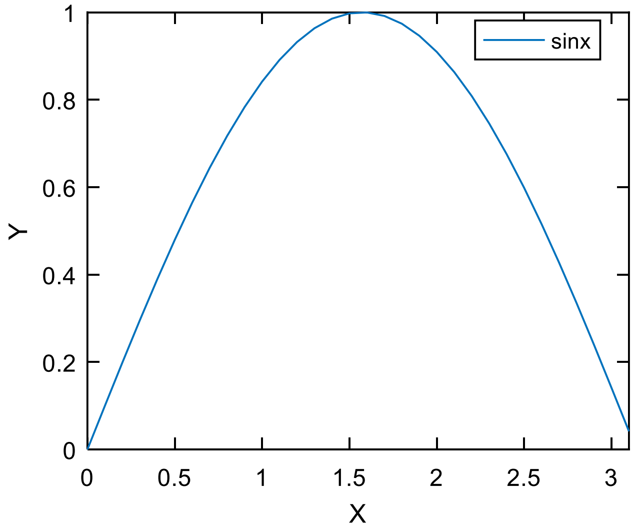
2. Then create a spectrum and name it “sinx”:

```
s = specdata(x,y,'sinx');
```

s *1x1 specdata*

3. Now plot the spectrum:

```
figure; plot(s)
```



>

CREATE SPECDATA ARRAYS

4. Create a new Y variable:

```
y = cos(x);
```

y *1x32 double*

1	0.99	0.98	0.96	0.92	…	-0.97	-1

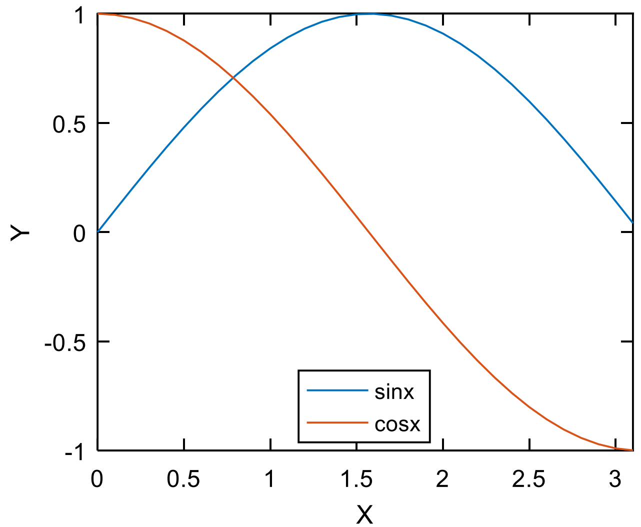
5. Create a second spectrum:

```
s(2) = specdata(x,y,'cosx');
```

s *1x2 specdata*

6. Plot both spectra at once:

```
figure; plot(s)
```



>

CREATE SPECDATA ARRAY FROM A MATRIX

1. Y is a matrix:

```
x = 0:0.1:pi;
y = [sin(x); cos(x)];
```

2. Create spectra:

```
s = specdata(x,y,{'sinx','cosx'});
```

3. Plot the spectra:

```
figure; plot(s)
```

>

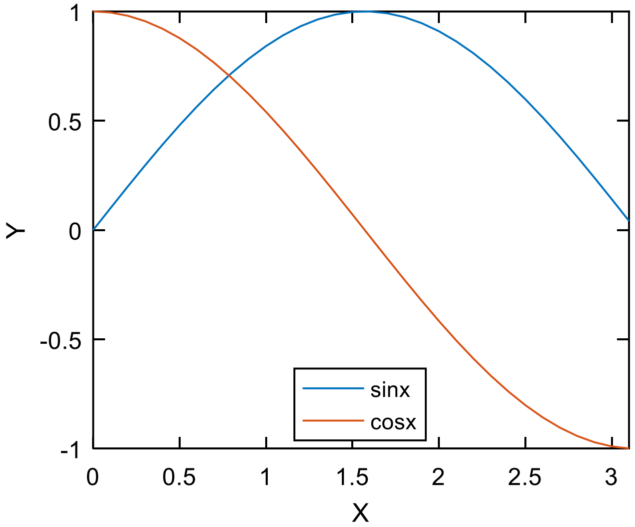
x *1x32 double*

0	0.1	0.2	0.3	0.4	…	3	3.1

y *2x32 double*

0	0.1	0.199	0.296	0.389	…	0.141	0.042
1	0.99	0.98	0.96	0.92	…	-0.97	-1

s *1x2 specdata*



LOAD DATA FROM A TEXT FILE

data1.txt

Wavelength	CD
350	13.428
350.5	13.247
351	13.041

…

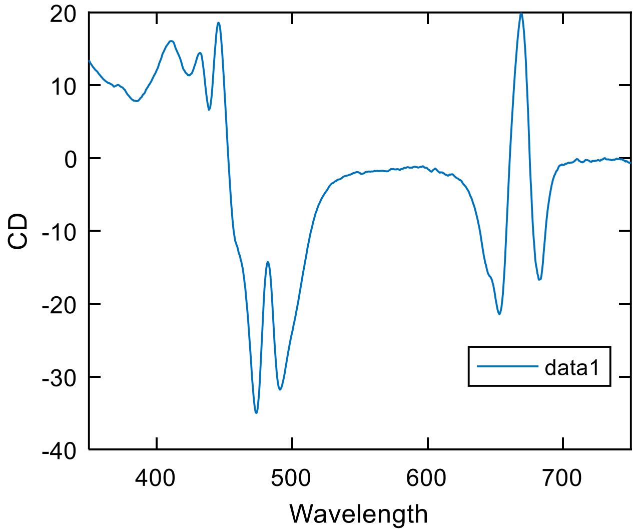
Load file

```
d = specdata.load('data1.txt');
```

Plot

```
figure; plot(d)
```

d *1x1 specdata*



>

LOAD MULTICOLUMN TEXT FILE

data2.txt

Wavelength	CD	HT	Abs
350	0.236	0.636	0.118
350.5	0.228	0.636	0.117
351	0.221	0.635	0.116

…

Load file

```
d = specdata.load('data2.txt');
```

Show properties ID, XType, YType

```
d.pt({'ID','XType','YType'})
```

d   *1x3 specdata*

```
   ID          XType         YType
_______   ______________    _____

'data2'   'Wavelength'      'CD'
'data2'   'Wavelength'      'HT'
'data2'   'Wavelength'      'Abs'
```

>

LOAD MULTIPLE FILES

data1.txt

Wavelength	CD
350	-7.226
350.5	-7.139
351	-7.097

…

data2.txt

Wavelength	CD
350	-8.592
350.5	-8.579
351	-8.533

…

data3.txt

Wavelength	CD
350	3.113
350.5	3.061
351	3.046

…

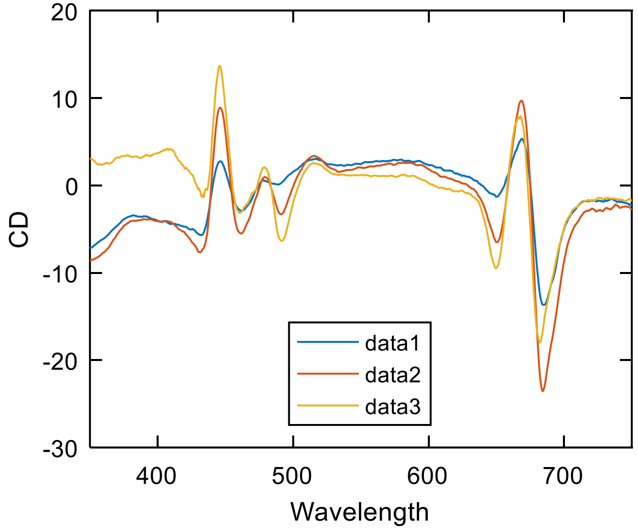
Load files

```
d = specdata.load('*.txt');
```

Plot

```
figure; plot(d)
```

d *3x1 specdata*



>

MULTIPLE MULTICOLUMN FILES

data1.txt

Wavelen	CD	Abs
350	-7.226	0.229
350.5	-7.139	0.227
351	-7.097	0.224

…

data2.txt

Wavelen	CD	Abs
350	-8.592	0.556
350.5	-8.579	0.552
351	-8.533	0.549

…

data3.txt

Wavelen	CD	Abs
350	3.113	0.500
350.5	3.061	0.498
351	3.046	0.495

…

Load files

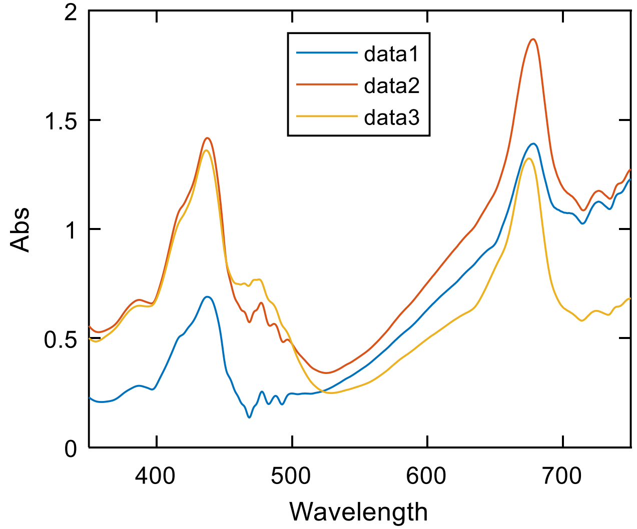
```
d = specdata.load('*.txt');
```

Plot 2nd column of *d*

```
figure; plot(d(:,2))
```

>

d *3x2 specdata*



OPERATE WITH SPECTRA

SPECTR-O-MATIC TOOLBOX

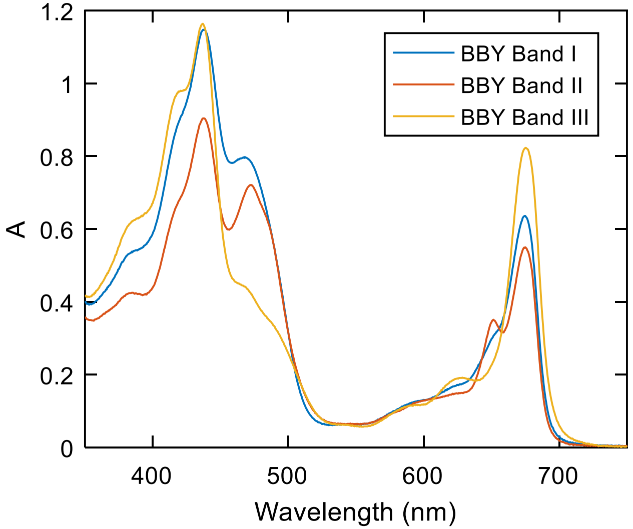
SCALAR OPERATIONS

Load data

```
load data.mat
```

```
>> whos
   Name   Size   Bytes   Class
   a      3x1    39090   specdata

>> plot(a)
```



Extract Y values (at position X)

```
m = a.Yx(675);
```

```
>> m'
   0.6360     0.5500     0.8230
```

Maximal values of Y (within a range)

```
mx = a.max([400 500])
```

```
>> mx'
   1.1480     0.9040     1.1640
```

Find peaks (of a minimum amplitude)

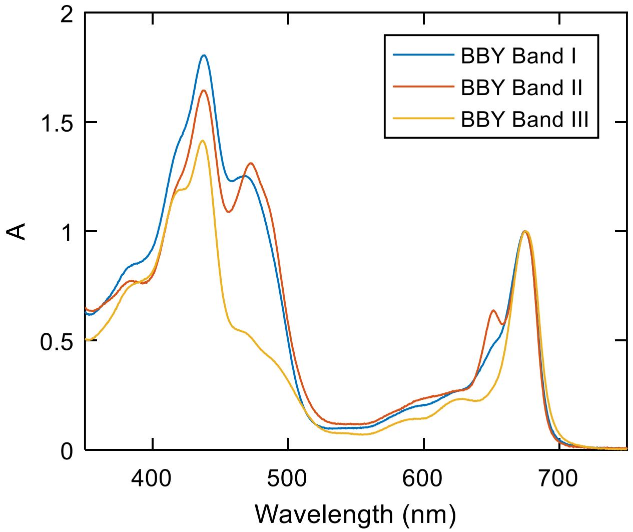
```
p = a(1).peaks(0.1)
```

```
>> p
   438.0000     1.1480
   674.5000     0.6360
```

OPERATIONS WITH SPECTRA AND SCALARS

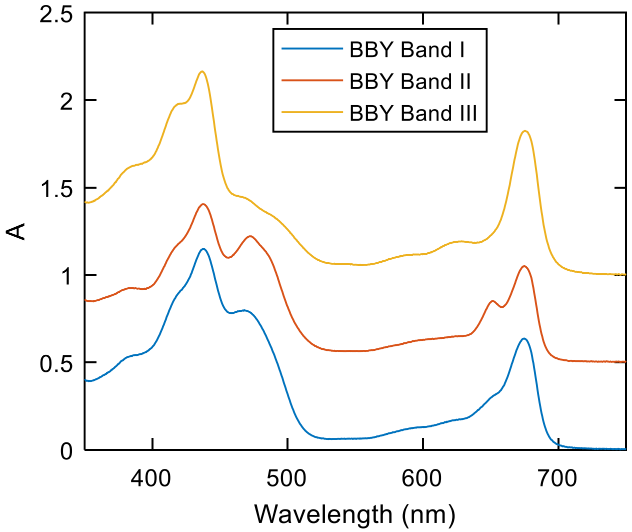
Divide spectra by scalar array

```
b = a / m;
```



Add scalar array to spectra

```
b = a + [0, 0.5, 1];
```



Shortcut

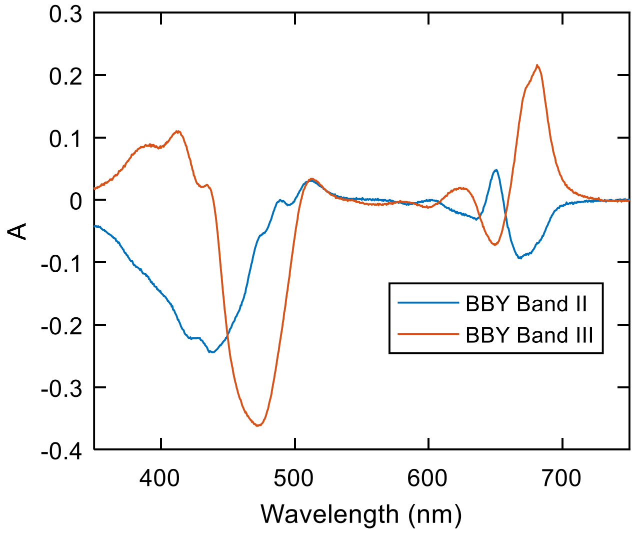
```
b = a.norm(675);
```

>

ARITHMETIC OPERATIONS ON SPECTRA

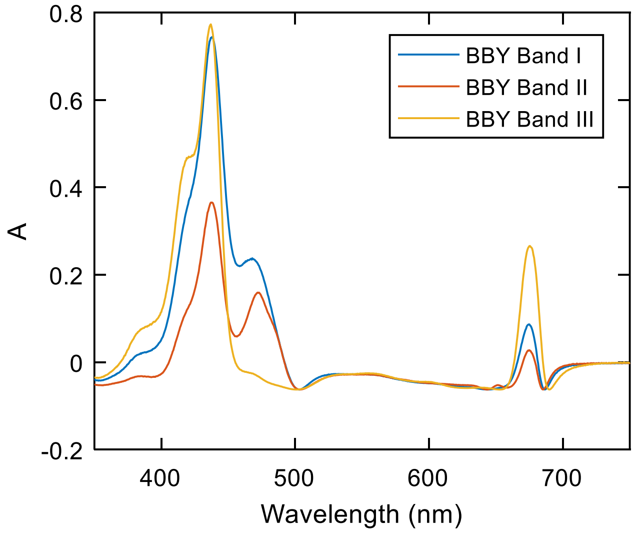
Subtract one spectrum from an array

```
b = a(2:3) – a(1);
```



Binary operations with arrays of spectra

```
b = -0.5*a + a^2;
```

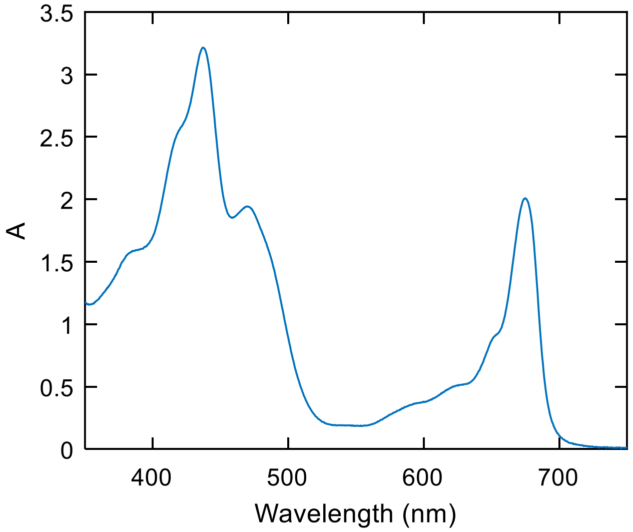


>

CALCULATE SUM AND MEAN

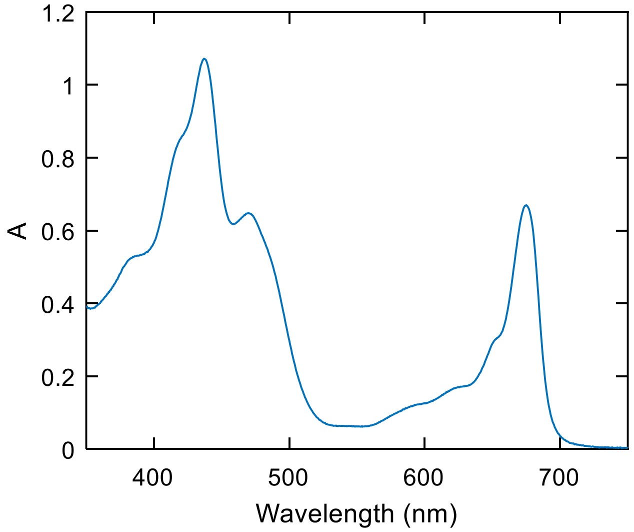
Sum spectra

```
b = sum(a);
```



Average spectra

```
b = mean(a);
```

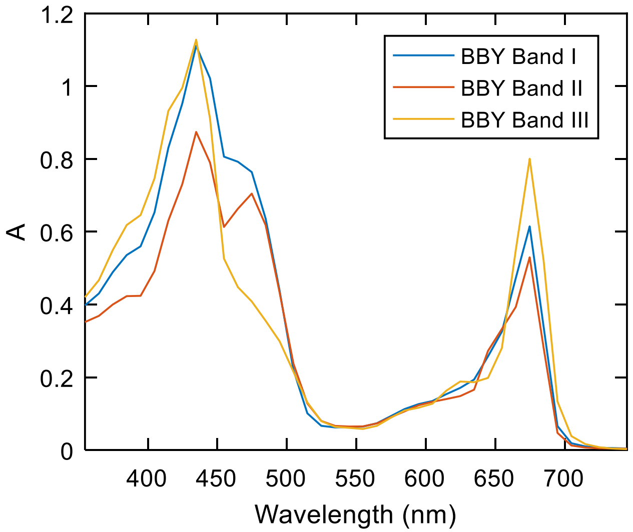


BIN AND SMOOTH

Bin data points (boxcar average)

```
b = a.bin(20);
```

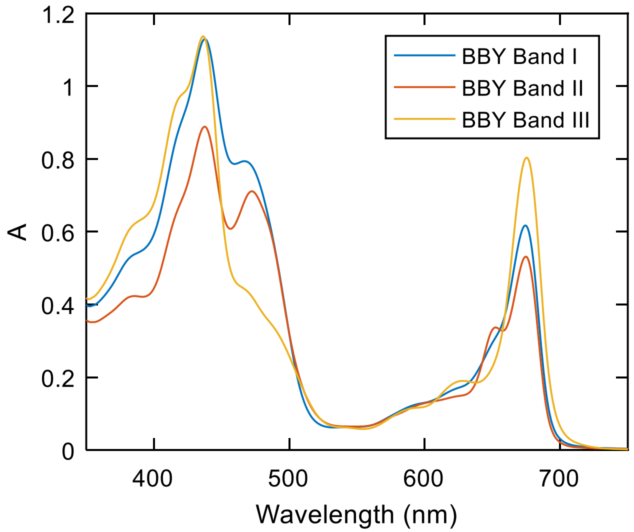
```
>> [b.dim]
   40    40    40
```



Smooth (moving average)

```
b = a.smooth(20);
```

```
>> [b.dim]
   801    801    801
```



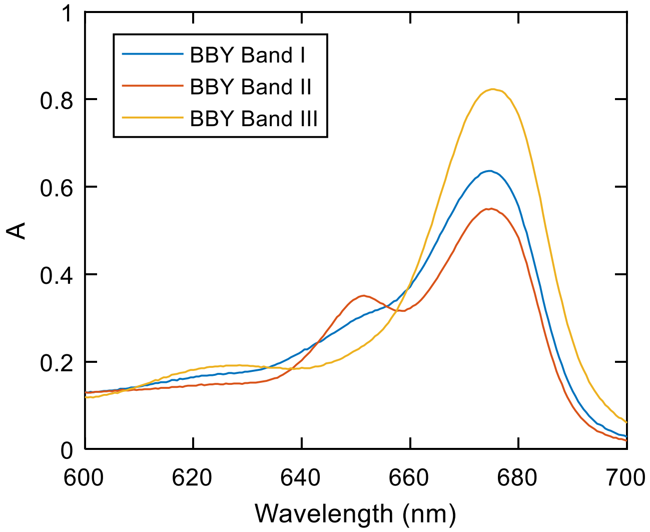
>

MANIPULATE X AXIS

Trim spectra (set X limits)

```
b = a.setxlim([600 700]);
```

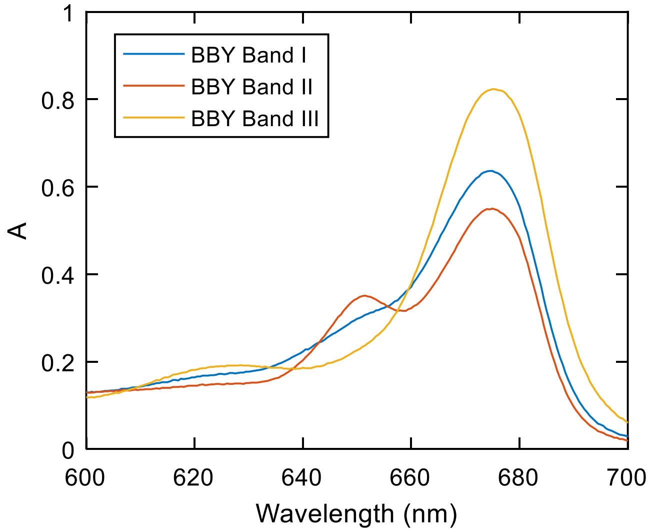
```
>> [b.dim]
   201    201    201
```



Change X axis (interpolate spectra)

```
b = a.setx(600:0.1:700);
```

```
>> [b.dim]
   1001    1001    1001
```

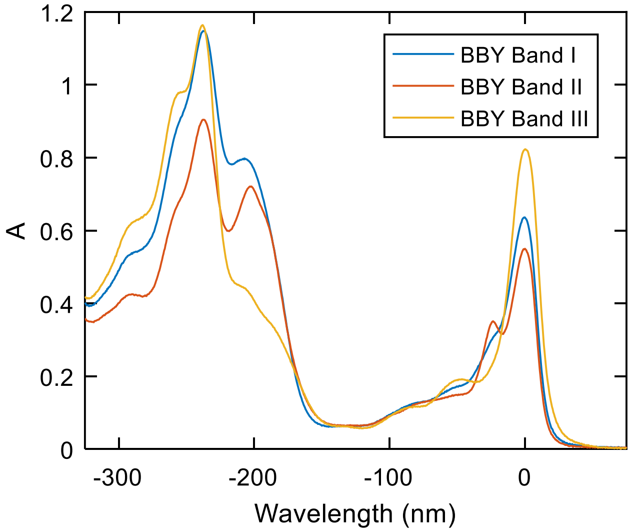


>

SHIFT AND JOIN SPECTRA

Shift spectra along X axis

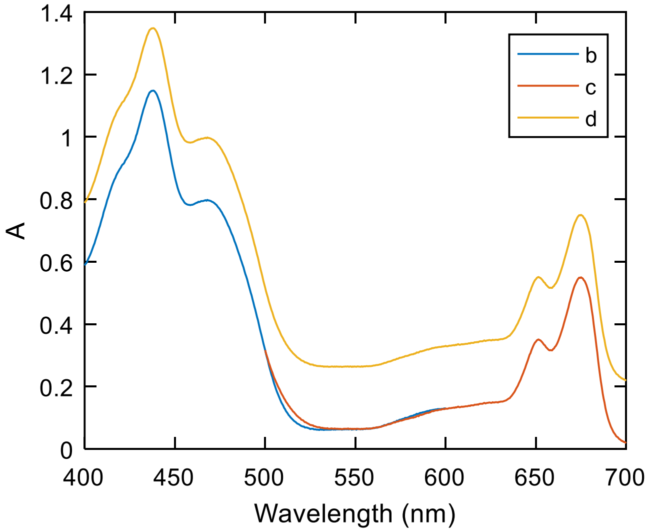
```
b = a.shiftx(-675);
```



Join spectra

```
b = a(1).setxlim([400 600]);
c = a(2).setxlim([500 700]);
d = merge([b,c]) + 0.5;
```

```
>> plot(b,c,d)
```



>

SELECT AND SEARCH

SPECTR-O-MATIC TOOLBOX

SELECT BY DIRECT INDEXING

Load data

```
load data.mat
```

```
>> whos
   Name   Size   Bytes    Class
   dat    3x3    117506   specdata

>> dat.get('ID')
   'SampleA fh'     'SampleA fv'     'SampleA iso'
   'SampleB fh'     'SampleB fv'     'SampleB iso'
   'blank fh'       'blank fv'       'blank iso'
```

Select a single spectrum

```
A = dat(1);
```

```
>> A.ID
   SampleA fh
```

Select multiple spectra

```
A = dat([1,4,7])
```

```
>> A.get('ID')
   'SampleA fh'     'SampleA fv'     'SampleA iso'
```

Select a row of spectra

```
A = dat(3,:)
```

```
>> A.get('ID')
   'blank fh'     'blank fv'     'blank iso'
```

>

SEARCH FOR KEYWORDS

Find spectra by ID

```
A = dat.find('SampleA');
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleA fv'
   'SampleA iso'
```

Search different properties

```
A = dat.find('ID','blank','YType','ABS');
```

```
>> A.pt('ID','YType')
         ID               YType
    'blank fh'        'ABSORBANCE'
    'blank fv'        'ABSORBANCE'
    'blank iso'       'ABSORBANCE'
```

Search for multiple keywords (match ANY)

```
A = dat.find({'SampleA','SampleB'});
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleB fh'
   'SampleA fv'
   'SampleB fv'
   'SampleA iso'
   'SampleB iso'
```

Search for multiple keywords (match ALL)

```
A = dat.find('ID','SampleA','ID','iso');
```

```
>> A.get('ID')
   'SampleA iso'
```

>

FIND INDEX

Find index of spectra

```
a = dat.findindex('SampleA');
```

```
>> whos a
   Name   Size   Bytes    Class
   a      9x1    9        logical
>> a'
   1      0      0      1      0      0      1      0      0
```

Select spectra using index

```
A = dat(a);
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleA fv'
   'SampleA iso'
```

Combine results (Boolean search)

```
a = dat.fi('SampleA');
b = dat.fi('SampleB');
ab  = a | b;  %  SampleA + SampleB
```

```
>> a'
   1      0      0      1      0      0      1      0      0
>> b'
   0      1      0      0      1      0      0      1      0
>> ab'
   1      1      0      1      1      0      1      1      0
```

```
i = dat.fi('iso');
abi = ab & i; % (SampleA + SampleB) * iso
```

```
>> i'
   0      0      0      0      0      0      1      1      1
>> abi'
   0      0      0      0      0      0      1      1      0
```

AUTOMATIC KEYWORD INDEX

Create an automatic keyword index

```
x = dat.autoindex;
```

```
>> x
   SampleA: [9x1 logical]
   SampleB: [9x1 logical]
     blank: [9x1 logical]
        fh: [9x1 logical]
        fv: [9x1 logical]
       iso: [9x1 logical]
```

Select spectra using index

```
A = dat(x.SampleA);
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleA fv'
   'SampleA iso'
```

Select spectra (Boolean OR)

```
A = dat(x.SampleA | x.SampleB);
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleB fh'
   'SampleA fv'
   'SampleB fv'
   'SampleA iso'
   'SampleB iso'
```

Select spectra (Boolean AND)

```
A = dat(x.SampleA & x.iso);
```

```
>> A.get('ID')
   'SampleA iso'
```

>

ORGANIZE DATA BY CATEGORIES

SPECTR-O-MATIC TOOLBOX

CATEGORICAL ARRAYS

Load data

```
load data.mat
```

```
>> whos
   Name  Size  Bytes   Class
   dat   3x3   117506  specdata

>> dat.get('ID')
   'SampleA fh'    'SampleA fv'    'SampleA iso'
   'SampleB fh'    'SampleB fv'    'SampleB iso'
   'blank fh'      'blank fv'      'blank iso'
```

Create a categorical array (MATLAB 2016b+)

```
c = dat.catfind({'SampleA','SampleB'});
```

```
>> c
   9x1 categorical array
```

Select spectra using categorical array

```
A = dat(c=='SampleA');
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleA fv'
   'SampleA iso'
```

>

SPLIT-APPLY-COMBINE WORKFLOW

1. Create categorical array of samples

```
s = dat.catfind({'SampleA','SampleB'});
```

2. Average spectra per group

```
g = findgroups(s);
A = splitapply(@mean, dat, g);
```

```
>> A
   2×1 specdata array

% splitapply ignores <undefined> samples
```

3. Find baseline spectra

```
B = dat.find('blank');
```

4. Split data and subtract baselines per group

```
f = @(x) {x-B}; % function f(x) = {x-B}
A = splitapply(f, dat, g);
```

```
>> A
   2×1 cell array
         [3×1 specdata]
         [3×1 specdata]
```

5. Concatenate result (if needed)

```
A = cat(1, A{:});
```

```
>> A
   6×1 specdata array
```

>

CATEGORICAL INDEX

Define variables

```
vars = struct;
vars.Sample = {'SampleA','SampleB'};
vars.Code = {'iso','fh','fv'};
```

Create index of variables

```
C = dat.catindex(vars);
```

```
>> C
                         Sample        Code
     SampleA fh        SampleA        fh
     SampleA fv        SampleA        fv
     SampleA iso       SampleA        iso
     SampleB fh        SampleB        fh
     SampleB fv        SampleB        fv
     SampleB iso       SampleB        iso
     blank fh          <undefined>    fh
     blank fv          <undefined>    fv
     blank iso         <undefined>    iso
```

Select spectra

```
A = dat(C.Sample=='SampleA');
```

```
>> A.get('ID')
   'SampleA fh'
   'SampleA fv'
   'SampleA iso'
```

```
A = dat(isundefined(C.Sample));
```

```
>> A.get('ID')
   'blank fh'
   'blank fv'
   'blank iso'
```

```
A = dat(C.Sample=='SampleA' ...
        & C.Code=='iso');
```

```
>> A.get('ID')
   'SampleA iso'
```

>

SPLIT-APPLY OPERATIONS

Average spectra grouped by 'Sample'

```
A = dat.splitop(@mean, C, 'Sample');
```

```
>> A
   3×1 specdata array

% splitop treats <undefined> as a separate group
```

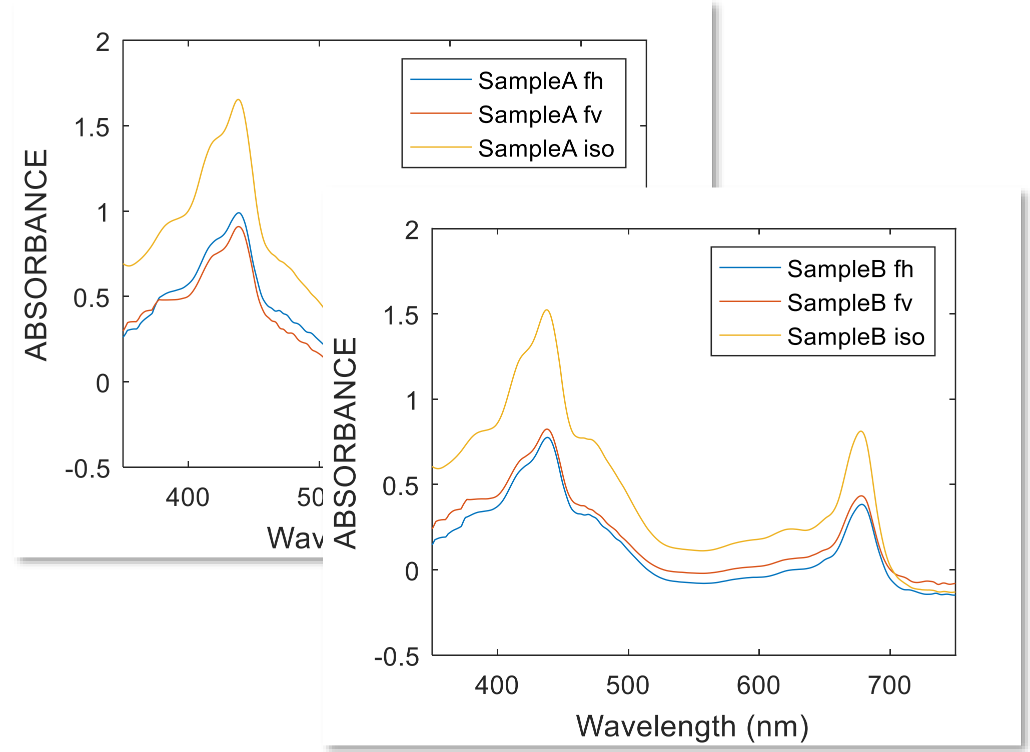
Subtract baselines grouping by 'Code'

```
b = dat.fi('blank');
A = dat.splitbinop(@minus,~b,b,C,'Code');
```

```
>> A
   1×6 specdata array
```

Plot spectra by sample group

```
C = A.catindex(vars);
A.splitop(@plotf, C, 'Sample');
```



>

CREATE PARAMETER TABLE

1. Create categorical index

```
C = A.catindex(vars);
```

```
>> C
                     Sample      Code
    SampleA fh       SampleA     fh
    SampleA fv       SampleA     fv
    SampleA iso      SampleA     iso
    SampleB fh       SampleB     fh
    SampleB fv       SampleB     fv
    SampleB iso      SampleB     iso
```

2. Add a calculated parameter column

```
C.Amax = A.max([600 700]);
```

```
 >> C
                     Sample      Code        Amax
    SampleA fh       SampleA     fh          0.52855
    SampleA fv       SampleA     fv          0.51194
    ...
```

3. Pivot table with 'Code' in columns

```
P = unstack(C,'Amax','Code');
```

```
>> P
   Sample        iso        fh         fv
   SampleA     1.0947     0.5286     0.5119
   SampleB     0.8120     0.3831     0.4333
```

4. Pivot table with 'Sample' in columns

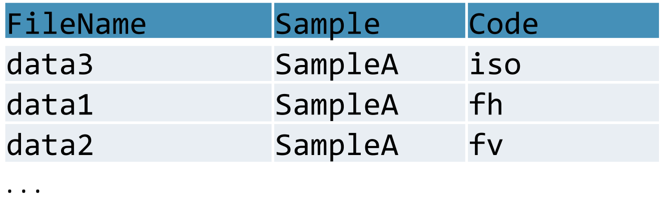
```
P = unstack(C,'Amax','Sample');
```

```
>> P
    Code        SampleA      SampleB
    fh          0.52855      0.38317
    fv          0.51194      0.43329
    iso         1.0947       0.81202
```

>

USE EXTERNAL CATEGORICAL INDEX

DataIndex.xls



FileName	Sample	Code
data3	SampleA	iso
data1	SampleA	fh
data2	SampleA	fv
...		

Load data files

```
dat = specdata.load('data*.txt');
```

```
>> dat.get('ID')
   'data1'
   'data2'
   'data3'
   ...
```

Synchronize external index table to data

```
C = indextable({dat.ID},'DataIndex.xls');
```

```
>> C
                 Sample     Code
    data1        SampleA    fh
    data2        SampleA    fv
    data3        SampleA    iso
    ...
```

Synchronize preloaded index table to data

```
T = readtable('DataIndex.xls', ...
              'ReadRowNames',true);
C = A.indextable({dat.ID}, T);
```

>