kpp_vs_kppa

Unknown Author

May 27, 2015

In [13]:
```
kpp_file_1 = 'extended_box_model_Exa2Green/box_model.dat_OpenMP'
kpp_file_2 = 'extended_box_model_Exa2Green/kpp-2.2.3.dat'
```

In [14]:
```
%matplotlib inline
import re
from itertools import cycle
from pylab import *
from matplotlib.markers import MarkerStyle
import matplotlib.pyplot as plt

ATOL = 1.0e-2
RTOL = 1.0e-2
EPS = 2.2204460492503131E-016
REGEX = re.compile('^([+\-]?)([0-9.]+)e?([+\-])([0-9.]+)$')
def convert(s):
    """
    Converts a number in Fortran E24.16 format to a Python float
    """
    m = re.search(REGEX, s)
    if m:
        s = ''.join([m.group(1), m.group(2), 'e', m.group(3), m.group(4)])
    try:
        fval = float(s)
    except ValueError:
        print '================> %s' % s
        fval = 0.0
    if fval < EPS:
        return 0.0
    else:
        return fval

def read_datfile(fname, tstart, cstart):
    """
    Read data from fname beginning on line tstart with concentration data beginning in
    Returns a tuple: (time, concentrations)
    Time data:
    [t0 t1 ... tN]
    Concentration data:
    [ [SPC_0(t0) SPC_1(t0) ... SPC_N(t0)]
    [SPC_0(t1) SPC_1(t1) ... SPC_N(t1)]
    : : :
    [SPC_0(tN) SPC_1(tN) ... SPC_N(tN)] ]
    """
    t = []
    c = []
    with open(fname, 'r') as f:
        while tstart:
            f.readline()
            tstart -= 1
        for line in f:
            parts = line.split()
```

```
            t.append(convert(parts[0]))
            c.append([convert(x) for x in parts[cstart:]])
    return t, c

def plot_dat(data, xlabel='Time', ylabel='Conc', names=None, titles=None):
    """
    Draw a plot of data read from read_datfile
    """
    lines = ['-', '--', '-.', ':']
    markers = MarkerStyle.filled_markers
    linecycler = cycle(lines)
    markercycler = cycle(markers)
    datastyles = ['%s%s' % (linecycler.next(), markercycler.next()) for _ in data]
    ndat = len(data)
    nspec = len(data[0][1][0])
    x = data[0][0]
    for i in xrange(0, nspec):
        fig, ax = plt.subplots()
        for j, dat in enumerate(data):
            t, c = dat
            y = [ct[i] for ct in c]
            style = datastyles[j]
            if names:
                label = '%s' % names[j]
            else:
                label = '%d' % j
            ax.plot(x, y, style, label=label)
        if ndat > 1:
            ax.legend(loc=2)
        ax.set_xlabel(xlabel)
        ax.set_ylabel(ylabel)
        if titles:
            ax.set_title(titles[i])
        else:
            ax.set_title('Species %d' % i)
        show()

def scaled_err(x, y):
    if x or y:
        return abs(x-y)/max(x, y)
    elif x == y:
        return 0.0
    else:
        return float('inf')

def calc_err(d0, d1):
    c0 = d0[1]
    c1 = d1[1]
    err = []
    nsteps = len(c0)
    nspec = len(c0[0])
    sigPow = 0.0
    errPow = 0.0
    errCount = 0.0
    for i in xrange(0, nsteps):
        e = []
        for j in xrange(0, nspec):
            x = c0[i][j]
            y = c1[i][j]
            sigPow += x*x
            errPow += (x-y)*(x-y)
            serr = scaled_err(x,y)
            if serr > RTOL:
                print '%g > %g: %g, %g' % (serr, RTOL, x, y)
                errCount += 1
            e.append(serr)
```

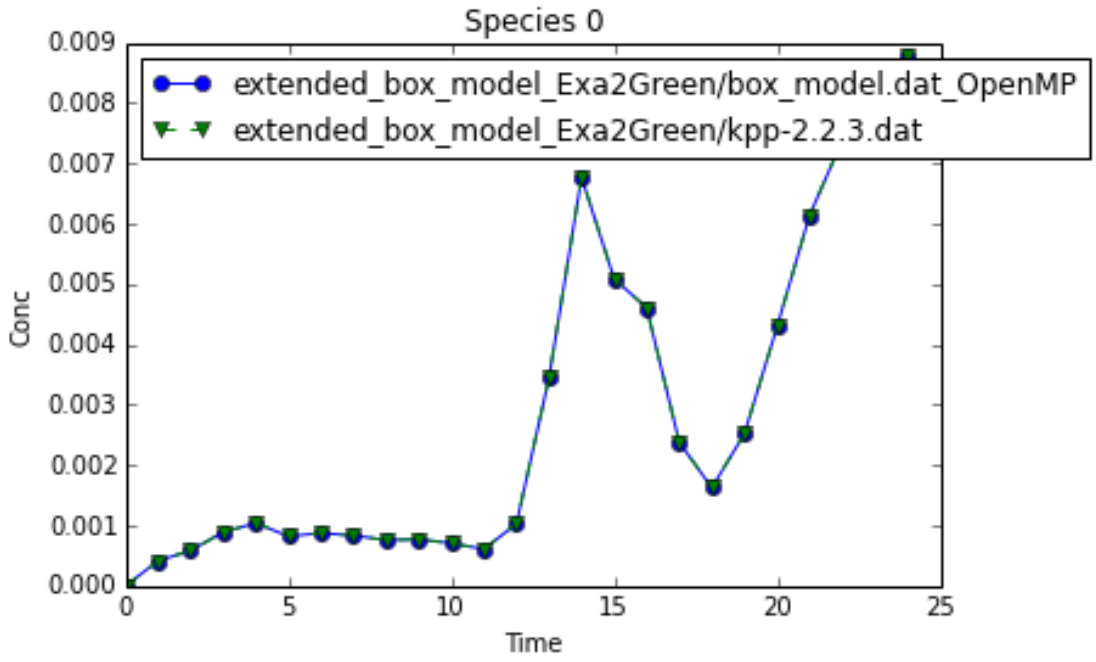
```
            err.append(e)
        if errPow > 0:
            snr = 20 * log10(sigPow / errPow)
        else:
            snr = float('inf')
        print 'SNR: %fdb' % snr
        if errCount:
            print '%d samples with relative error > %g' % (errCount, RTOL)
        return d1[0], err
```

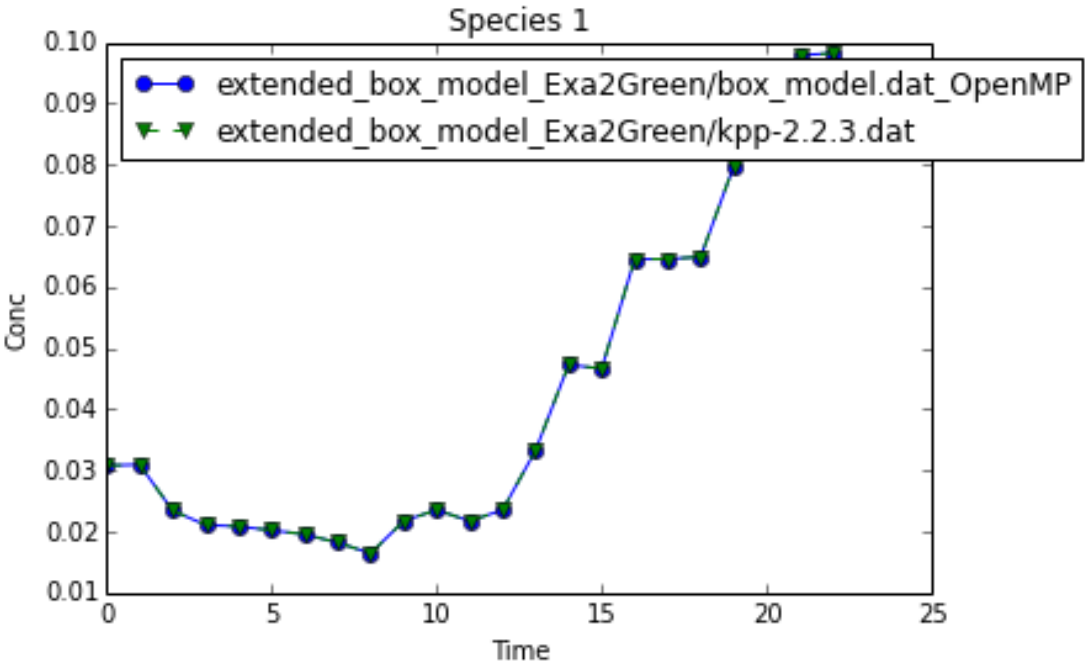
In [15]:
```
kpp_dat_1 = read_datfile(kpp_file_1, 0, 1)
kpp_dat_2 = read_datfile(kpp_file_2, 0, 1)
```

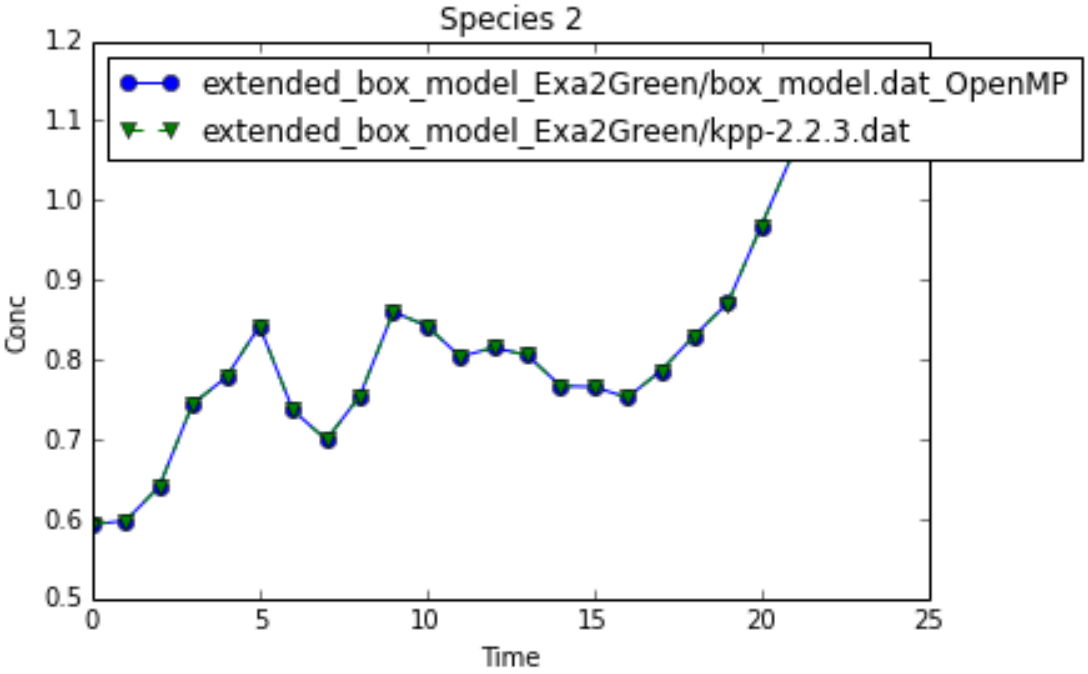
In [16]:
```
err_dat = calc_err(kpp_dat_1, kpp_dat_2)
```

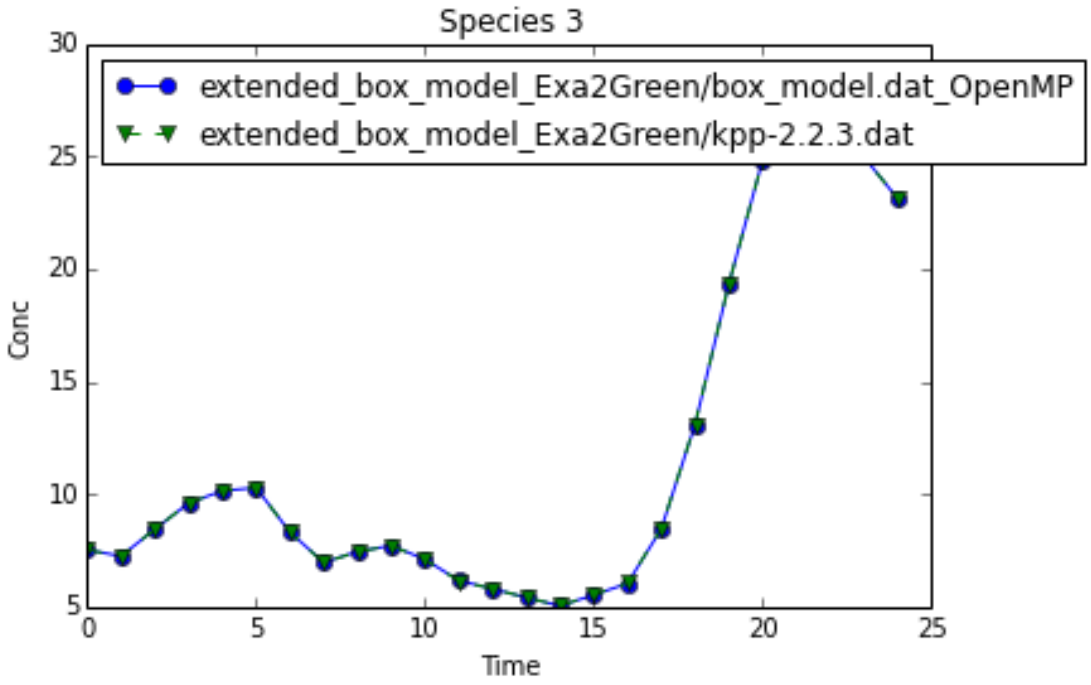
```
0.246924 > 0.01: 0.000202853, 0.000152764
0.0129232 > 0.01: 0.680091, 0.671302
0.0240016 > 0.01: 0.000199368, 0.000204271
0.0104154 > 0.01: 3.02862e-05, 2.99707e-05
0.0749483 > 0.01: 0.0195825, 0.021169
0.0102516 > 0.01: 0.000344305, 0.000340776
0.0148876 > 0.01: 3.04682e-07, 3.09286e-07
0.0122221 > 0.01: 0.00621802, 0.00629495
0.108972 > 0.01: 0.00024813, 0.000278476
SNR: 126.280031db
9 samples with relative error > 0.01
```

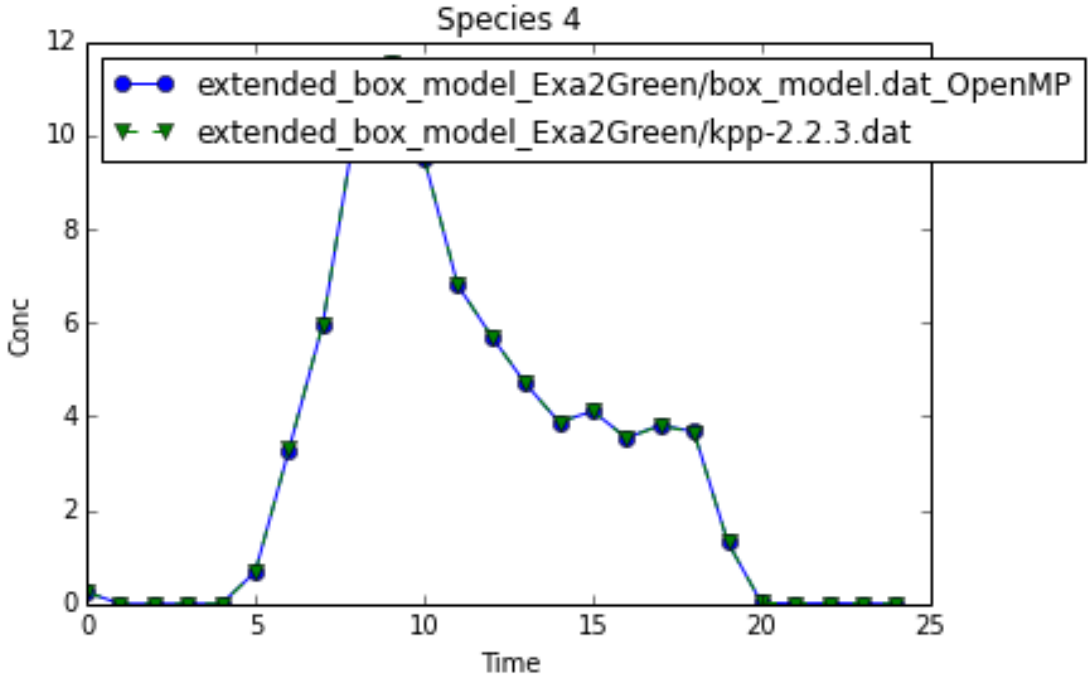
In [17]:
```
plot_dat([kpp_dat_1, kpp_dat_2], names=[kpp_file_1, kpp_file_2], titles=None)
```

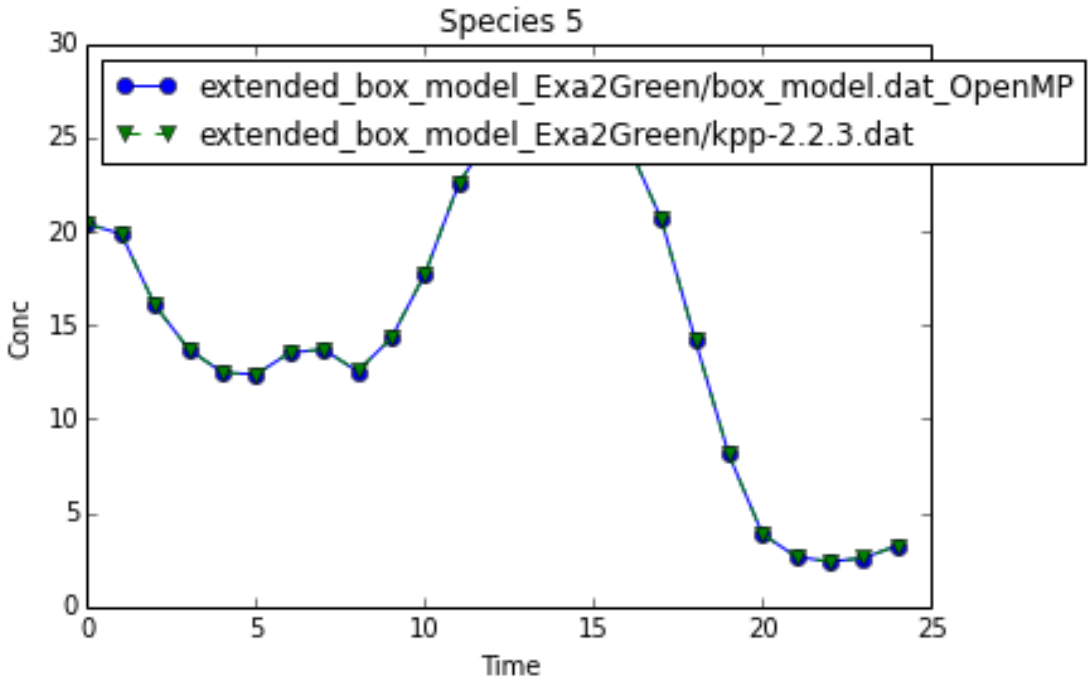


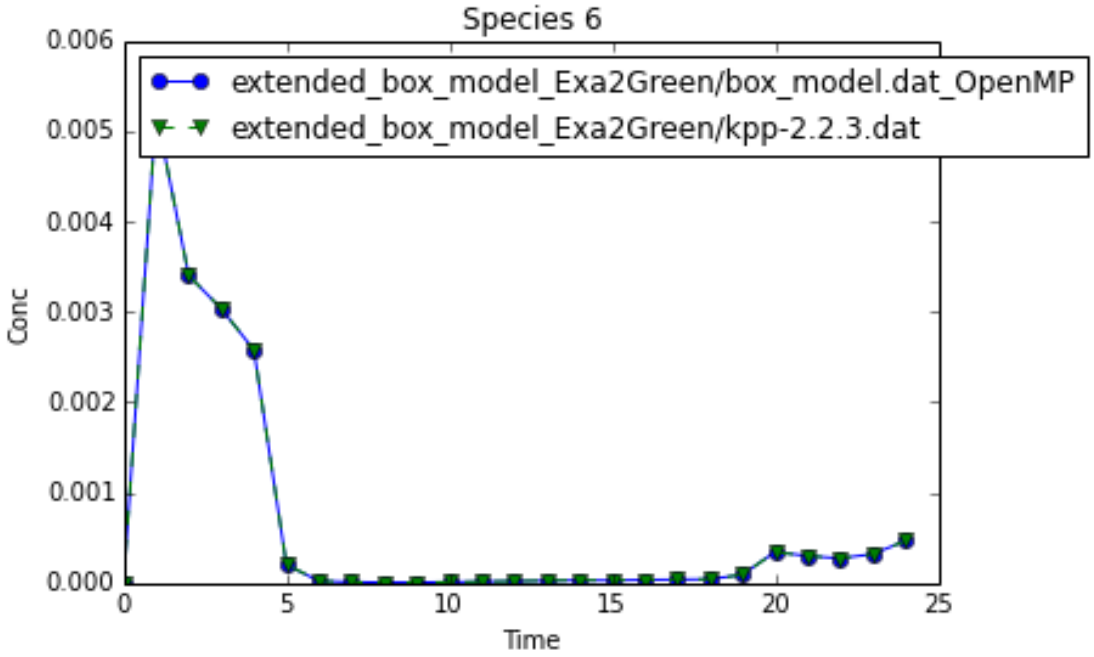

Species 1
extended_box_model_Exa2Green/box_model.dat_OpenMP
extended_box_model_Exa2Green/kpp-2.2.3.dat
Conc
Time
0.10
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
5
10
15
20
25

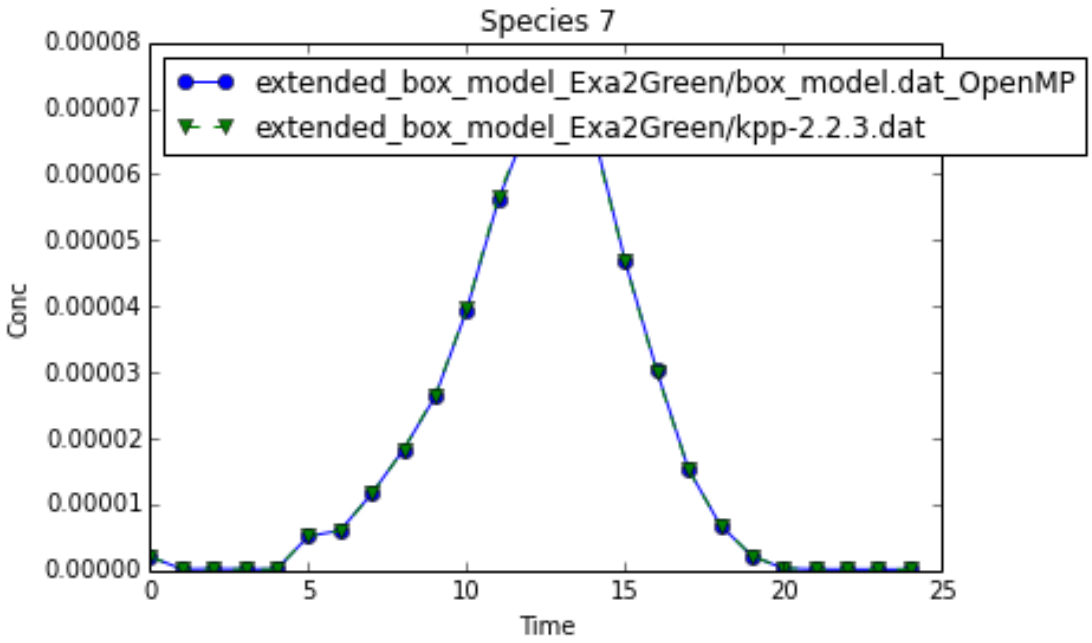

Species 2
extended_box_model_Exa2Green/box_model.dat_OpenMP
extended_box_model_Exa2Green/kpp-2.2.3.dat
Conc
Time
1.2
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0
5
10
15
20
25


Species 3
extended_box_model_Exa2Green/box_model.dat_OpenMP
extended_box_model_Exa2Green/kpp-2.2.3.dat
Conc
Time
30
25
20
15
10
5
0
5
10
15
20
25

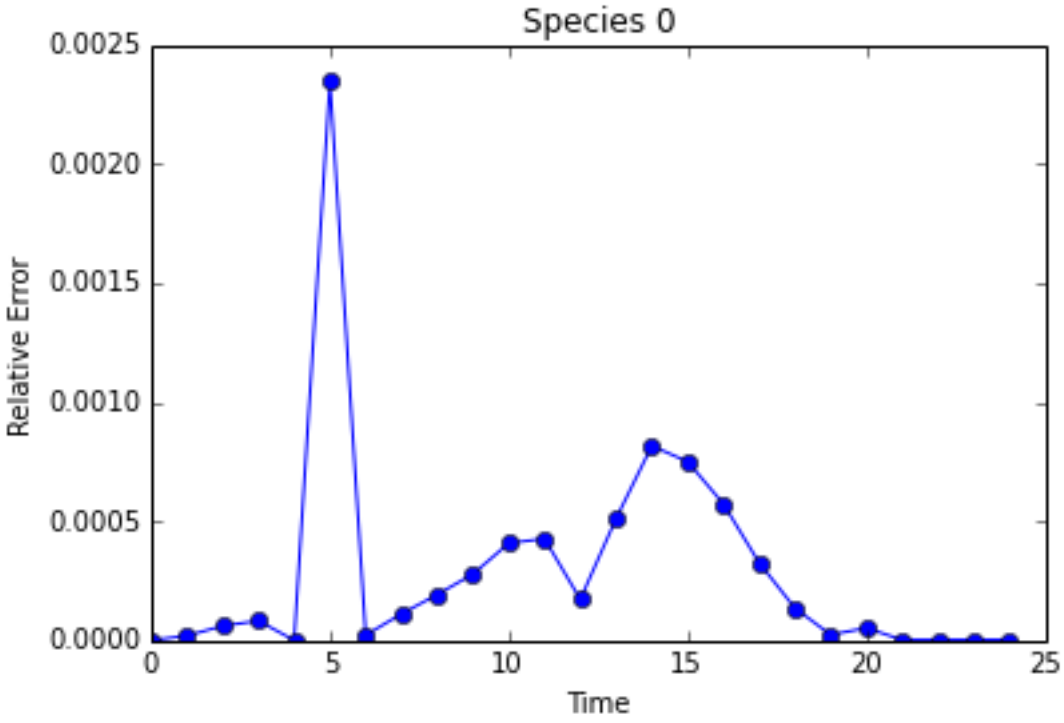

Species 4
extended_box_model_Exa2Green/box_model.dat_OpenMP
extended_box_model_Exa2Green/kpp-2.2.3.dat
Conc
Time
12
10
8
6
4
2
0
0
5
10
15
20
25

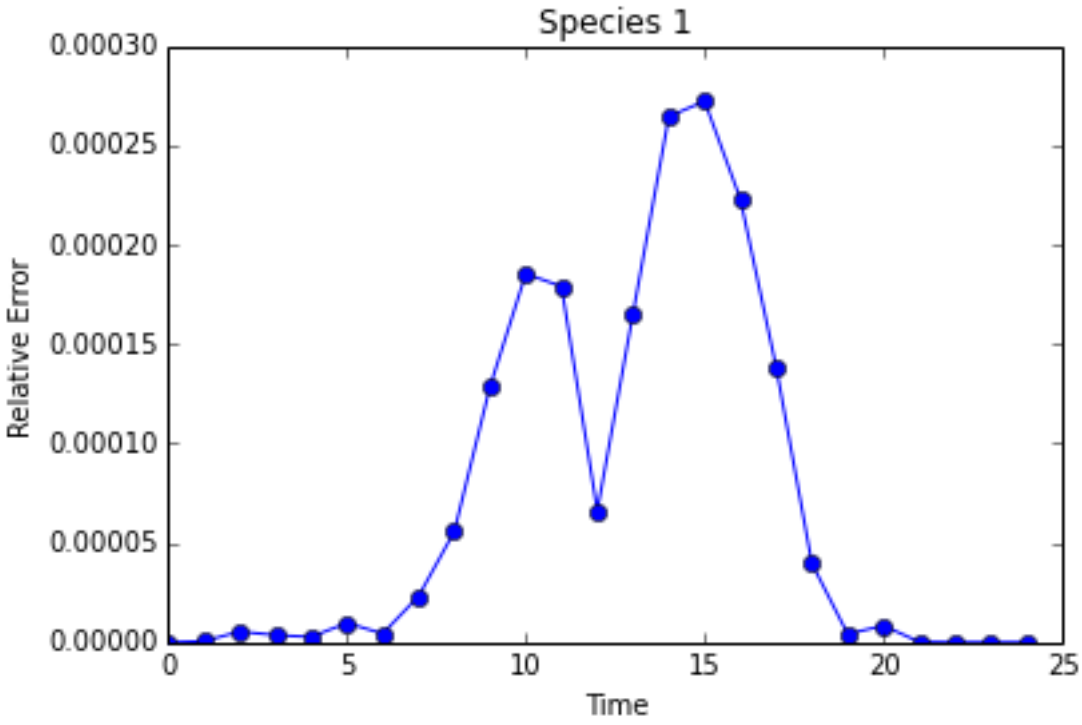

Species 5
extended_box_model_Exa2Green/box_model.dat_OpenMP
extended_box_model_Exa2Green/kpp-2.2.3.dat
Conc
Time
0
5
10
15
20
25
30

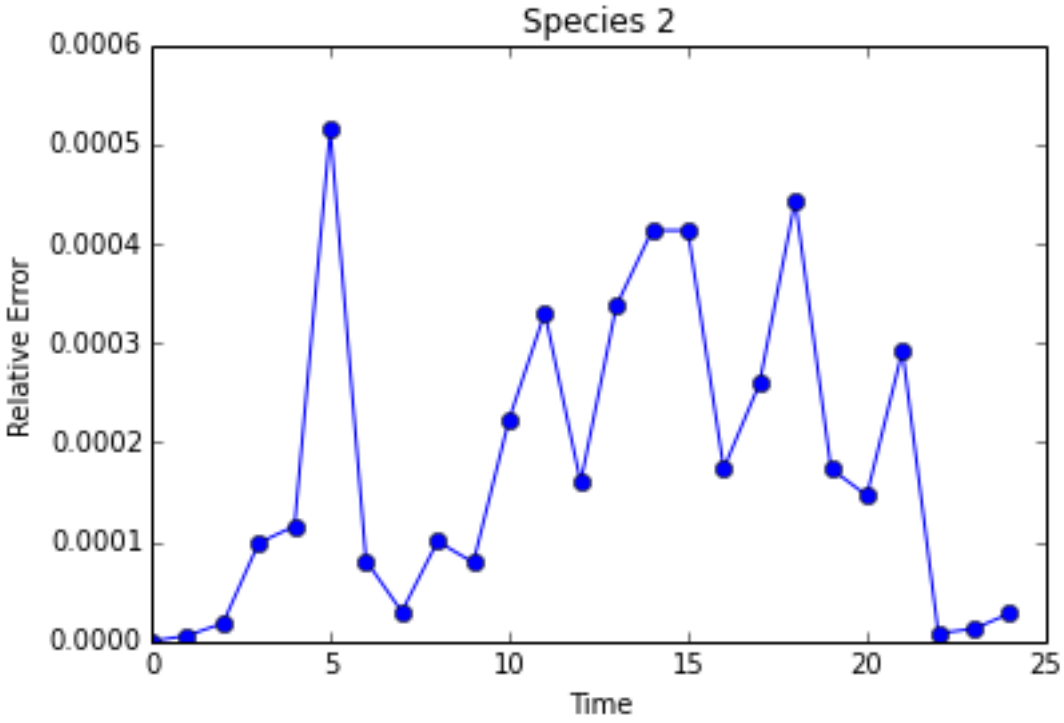

Species 6
extended_box_model_Exa2Green/box_model.dat_OpenMP
extended_box_model_Exa2Green/kpp-2.2.3.dat
Conc
Time
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0
5
10
15
20
25

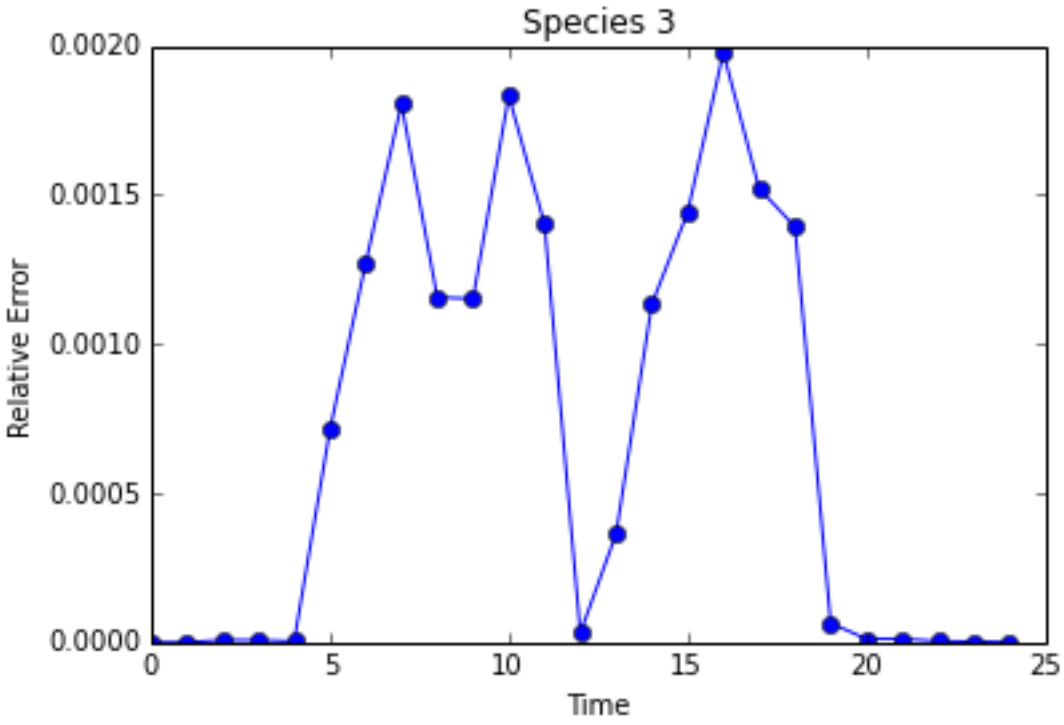


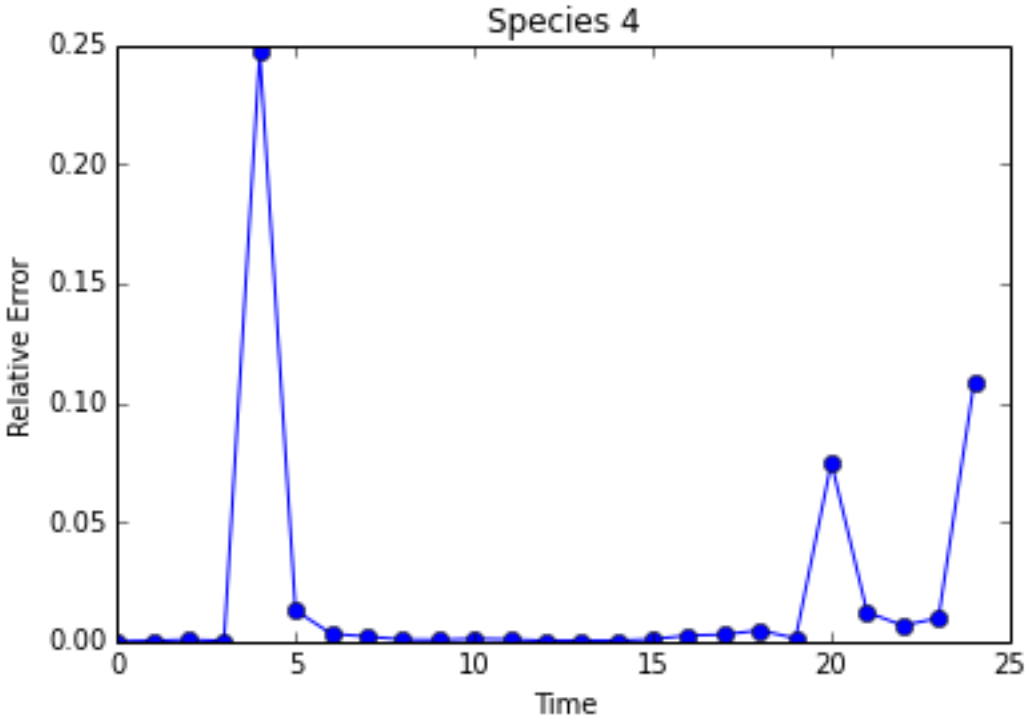
In [18]:
```
plot_dat([err_dat], ylabel='Relative Error')
```

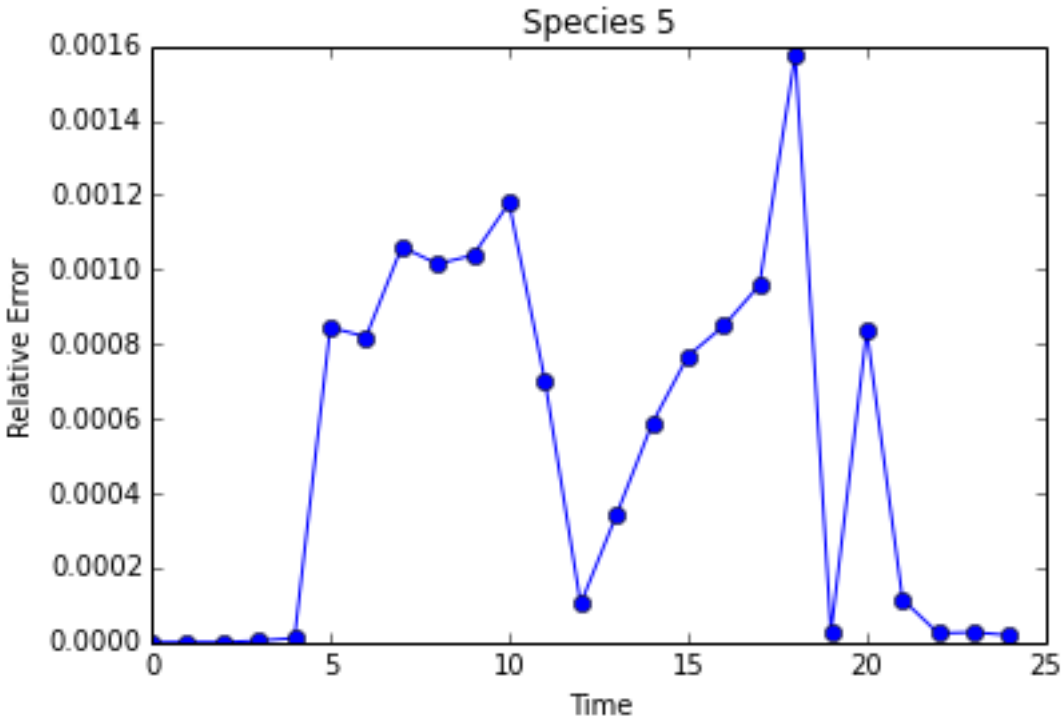


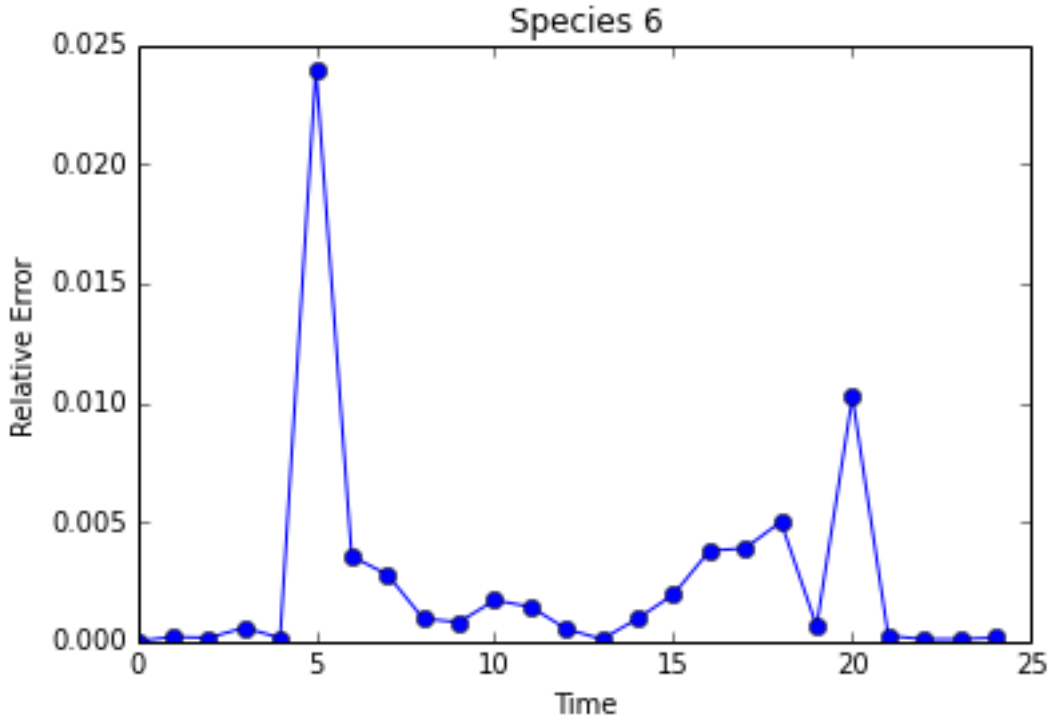

Species 1
Relative Error
Time
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
5
10
15
20
25

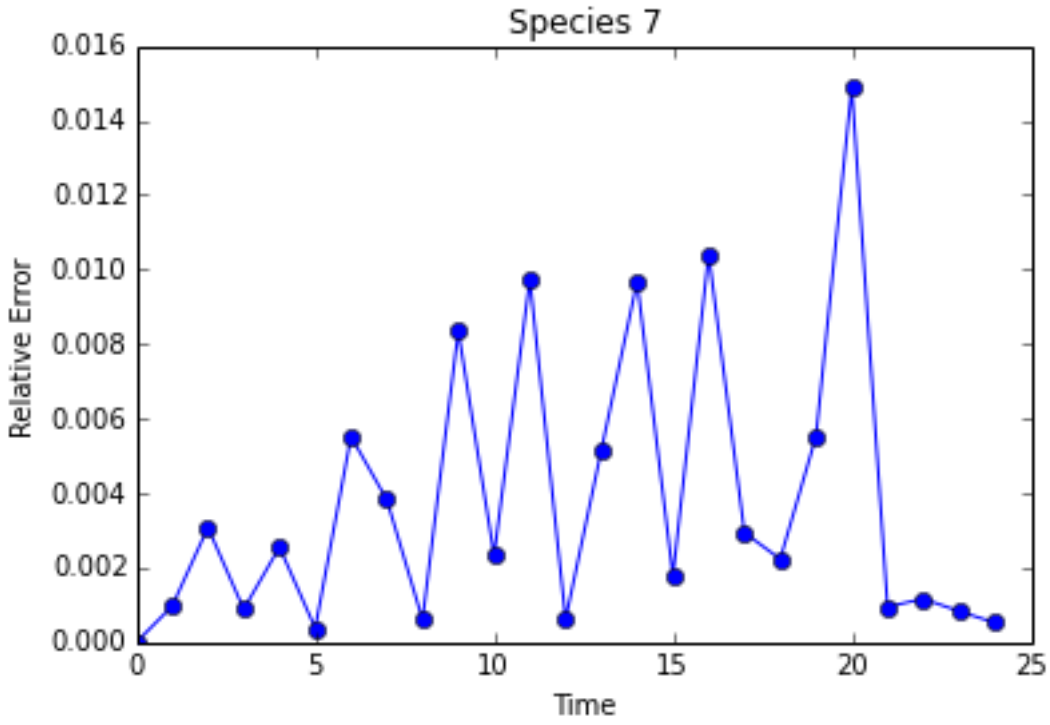

Species 2
Relative Error
Time
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
5
10
15
20
25


Species 3
Relative Error
0.0020
0.0015
0.0010
0.0005
0.0000
0
5
10
15
20
25
Time


Species 4
Relative Error
0.25
0.20
0.15
0.10
0.05
0.00
0
5
10
15
20
25
Time


Species 5
Relative Error
0.0016
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
5
10
15
20
25
Time


Species 6
Relative Error
0.025
0.020
0.015
0.010
0.005
0.000
0
5
10
15
20
25
Time


Species 7
Relative Error
Time
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
5
10
15
20
25

In [18]: