easydata

Unknown Author

December 11, 2013

In [1]:
```
%pylab inline
import sys
import glob
import os
import random

import numpy as np
import numpy.linalg as linalg
import numpy.random as rnd
from mpl_toolkits.mplot3d.axes3d import Axes3D
```

```
Populating the interactive namespace from numpy and matplotlib
```

Part I

Easydata GPLVM tests

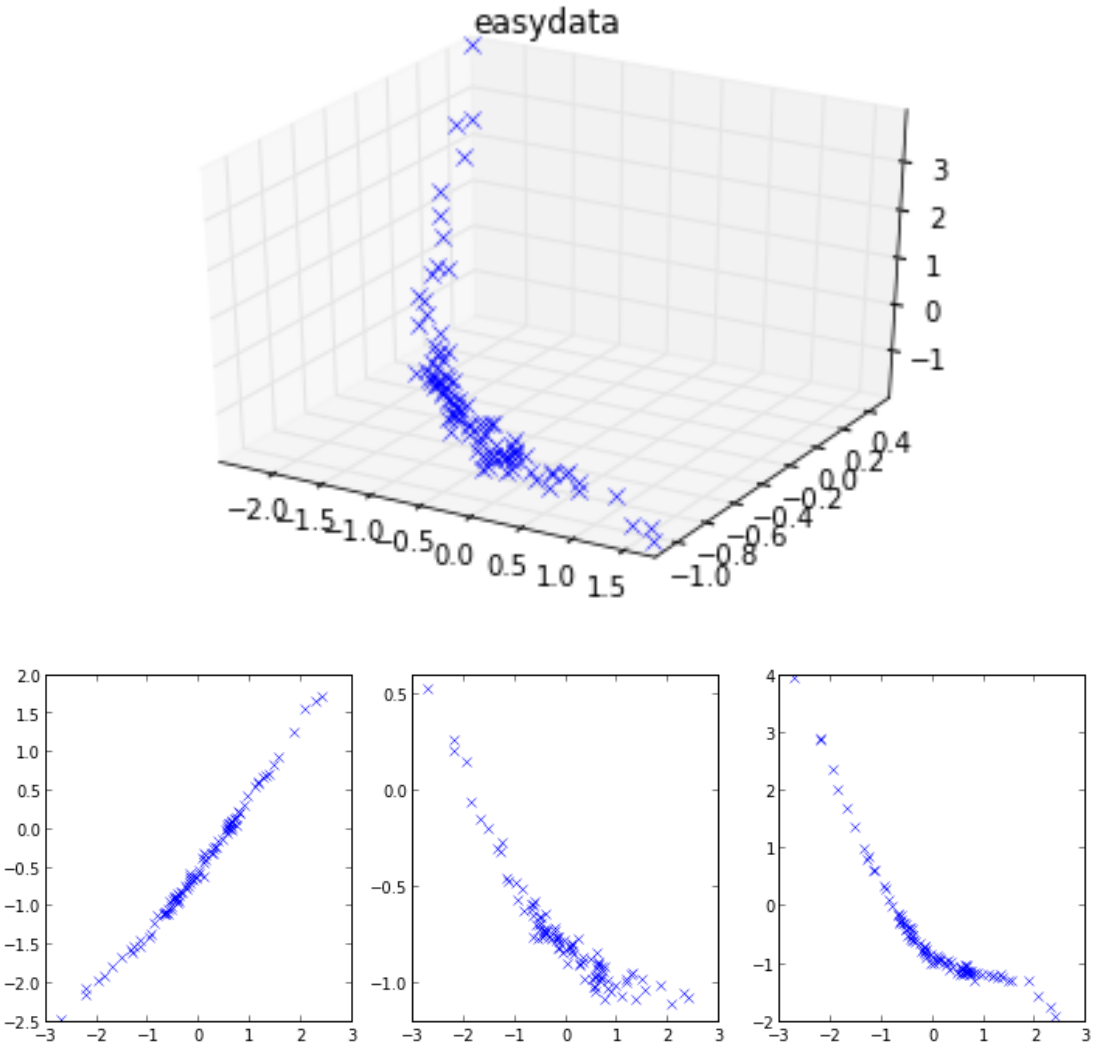
1 Generating the data

In [2]:
```
sys.path.append('./tools/')
import easy_dataset

Y, Xt = easy_dataset.gen_easydata(100, 1, 3)
```

After generating the data, plot in 3D and then each dimension as a function of the latent variable X:

In [3]:
```
fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot(Y[:, 0], Y[:, 1], Y[:, 2], 'x')
#ax.view_init(elev=60, azim=300)
ax.set_title('easydata')

fig, ax = plt.subplots(1, 3, figsize=(12, 4), dpi=180)
ax[0].plot(Xt, Y[:, 0], 'x')
ax[1].plot(Xt, Y[:, 1], 'x')
ax[2].plot(Xt, Y[:, 2], 'x')
```

Out [3]:
```
[<matplotlib.lines.Line2D at 0x4217a90>]
```



2 Initialisation with PCA

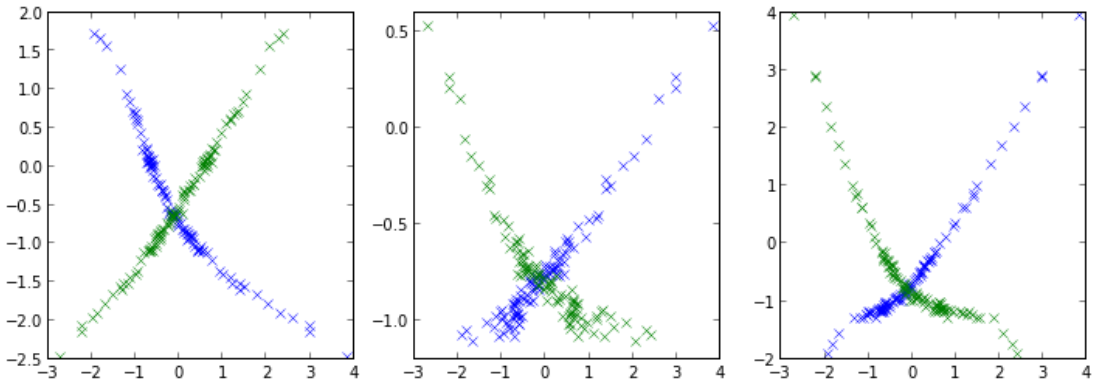
In [4]:
```
def PCA(Y, input_dim):
Z = numpy.linalg.svd(Y - Y.mean(axis=0), full_matrices=False)
[X, W] = [Z[0][:, 0:input_dim], numpy.dot(numpy.diag(Z[1]), Z[2]).T[:, 0:i
v = X.std(axis=0)
X /= v;
W *= v;
return X, W

X, W = PCA(Y, 2)
```

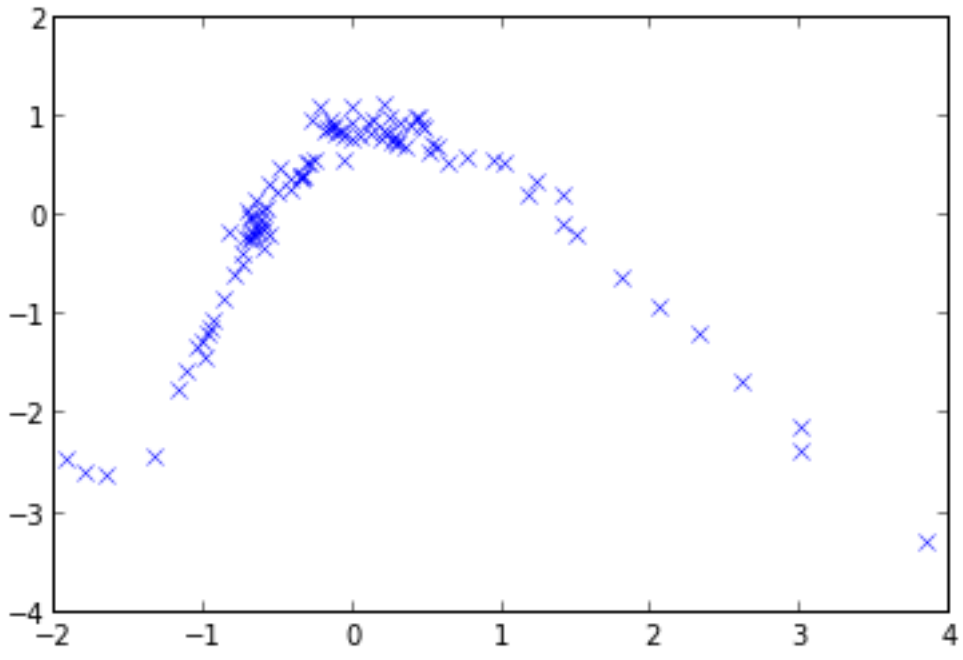
The main principle component of PCA manages to recover the correct latent coordinates (with perhaps a horizontal flip). In any case, the ordering is correctly found, which should give a very good initialisation for the GPLVM.

In [5]:
```
fig, ax = plt.subplots(1, 3, figsize=(12, 4), dpi=180)
ax[0].plot(X[:, 0], Y[:, 0], 'x')
ax[0].plot(Xt, Y[:, 0], 'x')
```

```
ax[1].plot(X[:, 0], Y[:, 1], 'x')
ax[1].plot(Xt, Y[:, 1], 'x')
ax[2].plot(X[:, 0], Y[:, 2], 'x')
ax[2].plot(Xt, Y[:, 2], 'x')

plt.figure()
plt.plot(X[:, 0], X[:, 1], 'x')
```

Out [5]: [<matplotlib.lines.Line2D at 0x46b9dd0>]





We can also find the marginal likelihood of the model:

In [6]:
```
import MLtools
# Requires optimisation over beta, so maybe later.
```

3 GPy GPLVM results

We now run the GPy Bayesian GPLVM to see if sensible results are obtained. Sheffield seem to fix the noise while performing optimisation over (presumably) X, ard, Z etc.

In [7]:
```
# Parameters to adjust
Q = 2
num_inducing = 10
```

In [8]:
```
import GPy
np.random.seed(0)

# Normalise data
Yn = Y - Y.mean(0)
Yn /= Yn.std(0)

# Set up model
rbf_comp = GPy.kern.rbf(Q, ARD=True)
kern = rbf_comp + GPy.kern.bias(Q, np.exp(-2)) + GPy.kern.white(Q, np.exp
m = GPy.models.BayesianGPLVM(Y, Q, kernel=kern, num_inducing=10)
m['.*lengt'] = 1.                # ???
m['noise'] = Yn.var() / 100
m.ensure_default_constraints()

m.constrain_fixed('noise')
m.optimize('scg', messages=1, max_f_eval=100, gtol=.05)
print m.log_likelihood()
m.constrain_positive('noise')
m.optimize('scg', messages=1, max_f_eval=50, gtol=.05)
print m.log_likelihood()
```

```
Y is not zero mean, centering it locally (GPy.util.linalg.PCA)
Warning: re-constraining these parameters
noise_variance
 I      F              Scale          |g|
 0001   4.483407e+03   1.000000e+00   4.767712e+07  0002
2.489548e+03   5.000000e-01   4.461759e+06  0003   1.760196e+03
2.500000e-01   5.572565e+05  0004   1.147414e+03   1.250000e-01
9.910825e+04  0005   7.243977e+02   6.250000e-02   2.501907e+04  0006
6.582098e+02   3.125000e-02   3.024987e+04  0007   5.541209e+02
1.562500e-02   5.846255e+03  0008   4.868513e+02   7.812500e-03
4.493099e+03  0009   4.522892e+02   3.906250e-03   3.853156e+03  0010
3.401087e+02   1.953125e-03   1.422380e+03  0011   3.206164e+02
9.765625e-04   2.431484e+03  0012   2.930142e+02   4.882812e-04
3.836857e+02  0013   2.822603e+02   2.441406e-04   6.409597e+02  0014
2.756054e+02   1.220703e-04   3.904500e+02  0015   2.693578e+02
6.103516e-05   7.100567e+02  0016   2.646521e+02   3.051758e-05
3.602441e+02  0017   2.601721e+02   1.525879e-05   4.394063e+02  0018
2.559853e+02   7.629395e-06   3.546968e+02  0019   2.518522e+02
3.814697e-06   4.465559e+02  0020   2.480109e+02   1.907349e-06
3.474832e+02  0021   2.442792e+02   9.536743e-07   3.945318e+02  0022
2.405357e+02   4.768372e-07   2.807561e+02  0023   2.365723e+02
2.384186e-07   5.376110e+02  0024   2.331927e+02   1.192093e-07
```

2.850710e+02 0025 2.297688e+02 5.960464e-08 4.445682e+02 0026
2.266241e+02 2.980232e-08 2.764947e+02 0027 2.235281e+02
1.490116e-08 3.952416e+02 0028 2.205760e+02 7.450581e-09
2.615552e+02 0029 2.177082e+02 3.725290e-09 3.773161e+02 0030
2.150274e+02 1.862645e-09 2.777283e+02 0031 2.124119e+02
9.313226e-10 3.291086e+02 0032 2.098987e+02 4.656613e-10
2.647292e+02 0033 2.074670e+02 2.328306e-10 3.086824e+02 0034
2.051232e+02 1.164153e-10 2.730318e+02 0035 2.028496e+02
5.820766e-11 2.884171e+02 0036 2.006457e+02 2.910383e-11
2.649391e+02 0037 1.985117e+02 1.455192e-11 2.751262e+02 0038
1.964294e+02 7.275958e-12 2.634591e+02 0039 1.944184e+02
3.637979e-12 2.528547e+02 0040 1.924579e+02 1.818989e-12
2.412362e+02 0041 1.905477e+02 9.094947e-13 2.579702e+02 0042
1.886803e+02 4.547474e-13 2.169411e+02 0043 1.868667e+02
2.273737e-13 2.547159e+02 0044 1.850761e+02 1.136868e-13
1.987904e+02 0045 1.833313e+02 5.684342e-14 2.676217e+02 0046
1.816640e+02 2.842171e-14 2.140383e+02 0047 1.800384e+02
1.421085e-14 2.318303e+02 0048 1.784520e+02 7.105427e-15
2.030302e+02 0049 1.769004e+02 3.552714e-15 2.296086e+02 0050
1.753863e+02 1.776357e-15 1.918012e+02 0051 1.739045e+02
8.881784e-16 2.263549e+02 0052 1.724630e+02 4.440892e-16
1.853929e+02 0053 1.710419e+02 2.220446e-16 2.271300e+02 0054
1.696717e+02 1.110223e-16 1.839101e+02 0055 1.683174e+02
5.551115e-17 2.176548e+02 0056 1.670049e+02 2.775558e-17
1.730298e+02 0057 1.656968e+02 1.387779e-17 2.214263e+02 0058
1.644402e+02 6.938894e-18 1.675985e+02 0059 1.632034e+02
3.469447e-18 2.068005e+02 0060 1.619252e+02 1.734723e-18
1.398641e+02 0061 1.606587e+02 8.673617e-19 2.438497e+02 0062
1.593152e+02 4.336809e-19 1.226274e+02 0063 1.579824e+02
2.168404e-19 2.869931e+02 0064 1.568221e+02 1.084202e-19
1.311685e+02 0065 1.556613e+02 5.421011e-20 2.322021e+02 0066
1.545138e+02 2.710505e-20 1.221044e+02 0067 1.533736e+02
1.355253e-20 2.389334e+02 0068 1.521390e+02 6.776264e-21
1.074650e+02 0069 1.508897e+02 3.388132e-21 2.933727e+02 0070
1.496965e+02 1.694066e-21 1.029500e+02 0071 1.485064e+02
8.470329e-22 2.844165e+02 0072 1.472155e+02 4.235165e-22
9.353928e+01 0073 1.459128e+02 2.117582e-22 3.374116e+02 0074
1.445660e+02 1.058791e-22 8.702137e+01 0075 1.432456e+02
5.293956e-23 3.575550e+02 0076 1.421872e+02 2.646978e-23
9.010402e+01 0077 1.411325e+02 1.323489e-23 2.707684e+02 0078
1.400373e+02 6.617445e-24 8.401790e+01 0079 1.389323e+02
3.308722e-24 3.000920e+02 0080 1.379092e+02 1.654361e-24
8.177409e+01 0081 1.368838e+02 8.271806e-25 2.800112e+02 0082
1.357825e+02 4.135903e-25 7.561640e+01 0083 1.346798e+02
2.067952e-25 3.194237e+02 0084 1.335896e+02 1.033976e-25
7.194078e+01 0085 1.324971e+02 5.169879e-26 3.259836e+02 0086
1.313452e+02 2.584939e-26 6.733881e+01 0087 1.301984e+02
1.292470e-26 3.578139e+02 0088 1.288393e+02 6.462349e-27
6.186405e+01 0089 1.274746e+02 3.231174e-27 4.550666e+02 0090
1.264506e+02 1.615587e-27 6.134087e+01 0091 1.254347e+02
8.077936e-28 3.299869e+02 0092 1.245048e+02 4.038968e-28
5.959619e+01 0093 1.235611e+02 2.019484e-28 3.099680e+02 0094
1.225629e+02 1.009742e-28 5.588947e+01 0095 1.215383e+02
5.048710e-29 3.522385e+02 0096 1.204506e+02 2.524355e-29
5.230255e+01 0097 1.193398e+02 1.262177e-29 3.985412e+02 0098

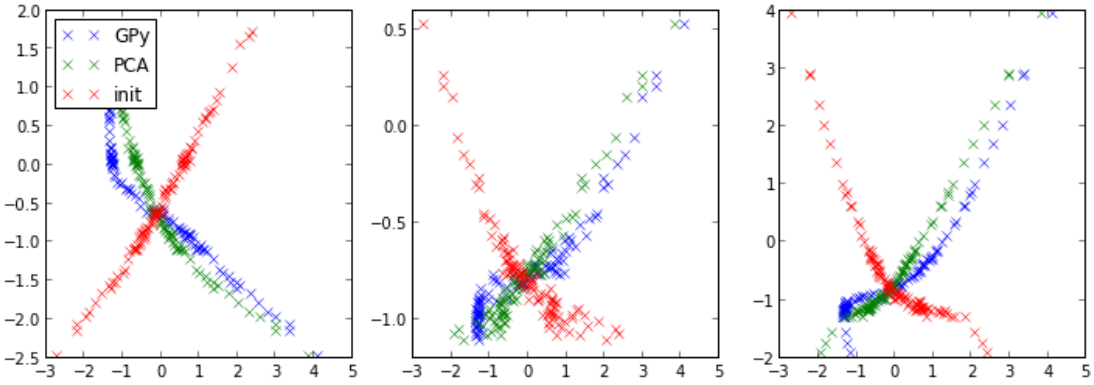
```
1.181818e+02   6.310887e-30   4.905750e+01  0098   1.181818e+02
3.155444e-30   4.318927e+02
-118.181773989
Warning: re-constraining these parameters
noise_variance
 I      F              Scale          |g|
 0001   1.155690e+02   1.000000e+00   9.307545e+02  0002
1.149024e+02   5.000000e-01   6.139673e+01  0003   1.143028e+02
2.500000e-01   9.817339e+01  0004   1.137900e+02   1.250000e-01
7.656325e+01  0005   1.133043e+02   6.250000e-02   6.856045e+01  0006
1.128582e+02   3.125000e-02   9.576625e+01  0007   1.124352e+02
1.562500e-02   5.978814e+01  0008   1.120230e+02   7.812500e-03
9.514012e+01  0009   1.116158e+02   3.906250e-03   5.519070e+01  0010
1.112162e+02   1.953125e-03   9.464615e+01  0011   1.108266e+02
9.765625e-04   5.304223e+01  0012   1.104394e+02   4.882812e-04
9.391983e+01  0013   1.100627e+02   2.441406e-04   5.064942e+01  0014
1.096975e+02   1.220703e-04   8.760531e+01  0015   1.093415e+02
6.103516e-05   5.005778e+01  0016   1.089925e+02   3.051758e-05
8.385606e+01  0017   1.086498e+02   1.525879e-05   4.848930e+01  0018
1.083103e+02   7.629395e-06   8.354468e+01  0019   1.079808e+02
3.814697e-06   4.725445e+01  0020   1.076563e+02   1.907349e-06
7.967860e+01  0021   1.073390e+02   9.536743e-07   4.613867e+01  0022
1.070254e+02   4.768372e-07   7.784678e+01  0023   1.067183e+02
2.384186e-07   4.486738e+01  0024   1.064172e+02   1.192093e-07
7.468020e+01  0025   1.061240e+02   5.960464e-08   4.473713e+01  0026
1.058364e+02   2.980232e-08   7.033787e+01  0027   1.055547e+02
1.490116e-08   4.441950e+01  0028   1.052764e+02   7.450581e-09
6.838918e+01  0029   1.050046e+02   3.725290e-09   4.400723e+01  0030
1.047362e+02   1.862645e-09   6.566952e+01  0031   1.044724e+02
9.313226e-10   4.327317e+01  0032   1.042129e+02   4.656613e-10
6.321641e+01  0033   1.039587e+02   2.328306e-10   4.337984e+01  0034
1.037066e+02   1.164153e-10   6.164608e+01  0035   1.034590e+02
5.820766e-11   4.271437e+01  0036   1.032147e+02   2.910383e-11
5.946631e+01  0037   1.029747e+02   1.455192e-11   4.265344e+01  0038
1.027382e+02   7.275958e-12   5.667085e+01  0039   1.025057e+02
3.637979e-12   4.295990e+01  0040   1.022758e+02   1.818989e-12
5.468734e+01  0041   1.020495e+02   9.094947e-13   4.302572e+01  0042
1.018244e+02   4.547474e-13   5.430439e+01  0043   1.016028e+02
2.273737e-13   4.215177e+01  0044   1.013822e+02   1.136868e-13
5.383139e+01  0045   1.011649e+02   5.684342e-14   4.131843e+01  0046
1.009491e+02   2.842171e-14   5.299531e+01  0047   1.007366e+02
1.421085e-14   4.104087e+01  0048   1.005262e+02   7.105427e-15
5.123594e+01  0048   1.005262e+02   3.552714e-15   4.096043e+01
-100.526181738
```

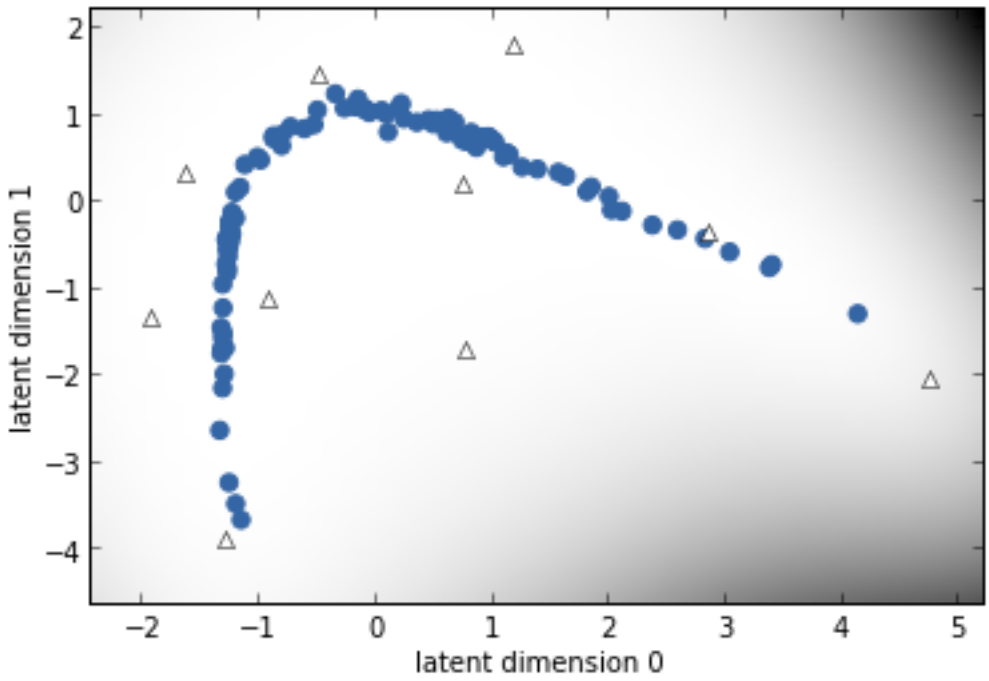
In [9]:

```
fig, ax = plt.subplots(1, 3, figsize=(12, 4), dpi=180)
ax[0].plot(m.X[:, 0], Y[:, 0], 'x', label='GPy')
ax[0].plot(X[:, 0], Y[:, 0], 'x', label='PCA')
ax[0].plot(Xt, Y[:, 0], 'x', label='init')
ax[0].legend(loc=2)

ax[1].plot(m.X[:, 0], Y[:, 1], 'x', label='GPy')
ax[1].plot(X[:, 0], Y[:, 1], 'x', label='PCA')
ax[1].plot(Xt, Y[:, 1], 'x', label='init')

ax[2].plot(m.X[:, 0], Y[:, 2], 'x', label='GPy')
```

```
ax[2].plot(X[:, 0], Y[:, 2], 'x', label='PCA')
ax[2].plot(Xt, Y[:, 2], 'x', label='init')

#fig, (latent_axes, sense_axes) = plt.subplots(1, 2)
#plt.sca(latent_axes)
plt.figure()
m.plot_latent()

plt.figure()
```

Out [9]: <matplotlib.figure.Figure at 0x67ae350>





<matplotlib.figure.Figure at 0x67ae350>

4 Parallel GPLVM results

This is a bit more difficult, as we need to copy the input data to a bunch of files.

In [29]:
```
P = 4
path = './easydata/'

# First delete all current inputs & embeddings
filelist = glob.glob("./easydata/inputs/*")
filelist.extend(glob.glob("./easydata/embeddings/*"))
for f in filelist:
    os.remove(f)

# Open files for writing the divided dataset into
f = []
for p in xrange(1, P + 1):
    name = path + 'inputs/easy_' + str(p)
    f.append(open(name, 'w'))

# Divide up dataset
for y in Y:
    x_str = ",".join(np.char.mod('%f', y))
    randf = random.choice(f)
    randf.write(x_str)
    randf.write('\n')

for fi in f:
    fi.close()
```

Now set up the options and call the actual script.

In [32]:
```
options = {}
options['input'] = './easydata/inputs/'
options['embeddings'] = './easydata/embeddings/'
options['parallel'] = 'local'
options['iterations'] = 10
options['statistics'] = './easydata/tmp'
options['tmp'] = './easydata/tmp'
options['M'] = num_inducing
options['Q'] = Q
options['D'] = 3
options['fixed_embeddings'] = False
options['keep'] = True

filelist = (glob.glob("./easydata/embeddings/*"))
for f in filelist:
    os.remove(f)

import parallel_GPLVM
parallel_GPLVM.main(options)
```

```
Creating ./easydata/embeddings//easy_2.embedding.npy with 23 points
Creating ./easydata/embeddings//easy_2.variance.npy with 23 points
Creating ./easydata/embeddings//easy_4.embedding.npy with 25 points
Creating ./easydata/embeddings//easy_4.variance.npy with 25 points
Creating ./easydata/embeddings//easy_1.embedding.npy with 29 points
Creating ./easydata/embeddings//easy_1.variance.npy with 29 points
Creating ./easydata/embeddings//easy_3.embedding.npy with 23 points
Creating ./easydata/embeddings//easy_3.variance.npy with 23 points
{'alpha': array([[ 1.,  1.]]), 'beta': array([[ 2.]]), 'Z':
array([[-0.33543584, -0.43848629],
       [ 1.44764714,  0.41104086],
       [-0.47384478, -0.18085939],
       [ 0.32714563,  1.07171832],
       [-0.58140747, -0.88636205],
       [ 0.2339834 ,  0.65243653],
```

```
       [-0.82322661, -0.32230887],
       [ 2.36102634, -1.32226271],
       [ 0.0148512 ,  1.02916016],
       [-0.0667572 ,  0.63208565]]), 'sf2': array([[ 1.]])}
Iteration  1
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  9  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[1] = -1445.08854038

{'alpha': array([[ 0.99866812,  0.99866812]]), 'beta': array([[
1.99733623]]), 'Z': array([[ -6.58498144e-01,  -4.70824348e-01],
       [  1.47746138e+00,   4.17350801e-01],
       [ -3.56682534e-01,  -5.60529219e-01],
       [ -8.96419213e-04,   1.01006124e+00],
       [ -5.02374363e-01,  -6.31417498e-01],
       [  3.54286857e-02,   7.00582407e-01],
       [ -7.47263533e-01,  -3.65288920e-01],
       [  2.36923548e+00,  -1.33382896e+00],
       [  3.30541799e-01,   7.26086731e-01],
       [  1.46122821e-01,   1.14853386e+00]]), 'sf2': array([[
0.99866812]])}
Iteration  2
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[2] = -1389.72481298

{'alpha': array([[ 0.96451243,  0.97040114]]), 'beta': array([[
1.8347759]]), 'Z': array([[-0.65352919, -0.43063426],
       [ 1.48947687,  0.42104361],
       [-0.45662788, -0.77317776],
       [ 0.06742194,  1.0437947 ],
       [-0.4145918 , -0.42251219],
       [-0.05655163,  0.73238215],
       [-0.76364403, -0.47378403],
       [ 2.3725682 , -1.33885387],
       [ 0.40668608,  0.68189257],
       [ 0.09893401,  1.19778325]]), 'sf2': array([[ 0.43485951]])}
Iteration  3
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
```

```
Done! embeddings Map-Reduce took  1  seconds
F[3] = -788.795237224

{'alpha': array([[ 0.93803735,  0.94960033]]), 'beta': array([[
1.70289246]]), 'Z': array([[-0.48450754, -0.47742274],
       [ 1.49163553,  0.42336696],
       [-0.51555833, -0.78879207],
       [ 0.11831669,  1.01292018],
       [-0.45040413, -0.37064028],
       [-0.04392009,  0.76569719],
       [-0.841332  , -0.46976168],
       [ 2.37336606, -1.34008999],
       [ 0.38023448,  0.71452386],
       [ 0.06314761,  1.16763445]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  4
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[4] = -506.005642624

{'alpha': array([[ 0.93796258,  0.94954279]]), 'beta': array([[
1.6920999]]), 'Z': array([[-0.48403798, -0.47763176],
       [ 1.49165029,  0.42337628],
       [-0.51566464, -0.78890293],
       [ 0.11831504,  1.01275977],
       [-0.45054884, -0.37045129],
       [-0.04391262,  0.76582345],
       [-0.84157301, -0.46970543],
       [ 2.37337086, -1.34009698],
       [ 0.38023875,  0.71453002],
       [ 0.06314017,  1.167732  ]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  5
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[5] = -490.672488281

{'alpha': array([[ 0.93766353,  0.94931263]]), 'beta': array([[
1.64892966]]), 'Z': array([[-0.48215971, -0.47846784],
       [ 1.49170935,  0.42341357],
       [-0.51608989, -0.78934638],
       [ 0.11830841,  1.01211814],
       [-0.45112769, -0.36969536],
       [-0.04388275,  0.76632849],
```

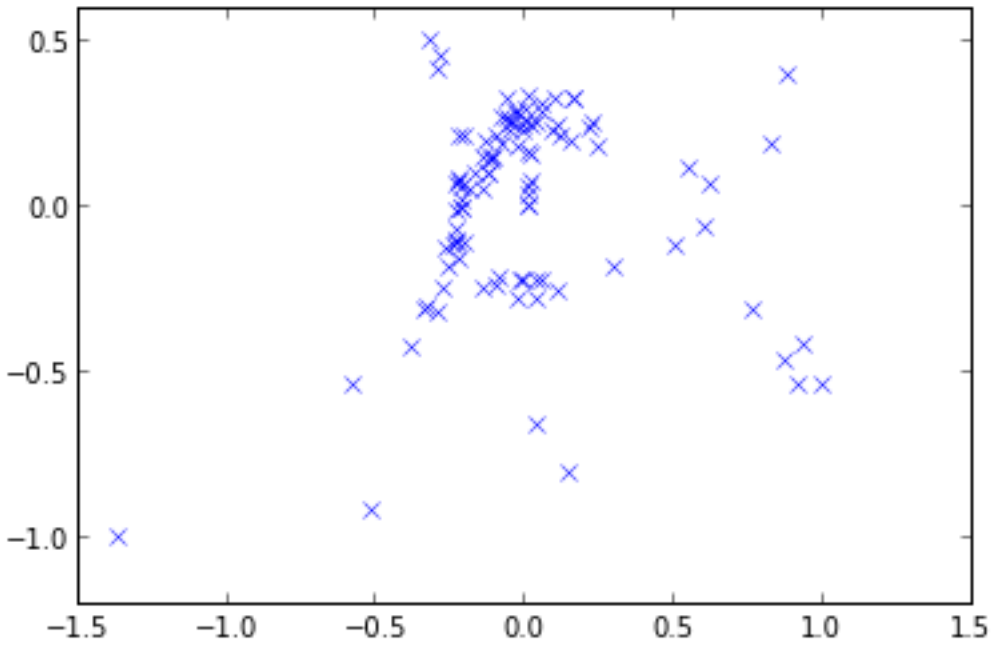
```
       [-0.84253705, -0.46948042],
       [ 2.37339005, -1.34012492],
       [ 0.38025582,  0.71455465],
       [ 0.06311044,  1.16812222]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  6
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[6] = -478.193811225

{'alpha': array([[ 0.93646729,  0.94839199]]), 'beta': array([[
1.47624872]]), 'Z': array([[-0.47464666, -0.48181215],
       [ 1.49194558,  0.42356273],
       [-0.51779089, -0.79112016],
       [ 0.1182819 ,  1.00955163],
       [-0.4534431 , -0.36667161],
       [-0.04376326,  0.76834868],
       [-0.84639323, -0.46858036],
       [ 2.37346682, -1.34023668],
       [ 0.3803241 ,  0.71465318],
       [ 0.06299148,  1.16968311]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  7
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[7] = -462.150060025

{'alpha': array([[ 0.93162132,  0.94466247]]), 'beta': array([[
0.77671249]]), 'Z': array([[-0.44421103, -0.49536008],
       [ 1.49290256,  0.42416699],
       [-0.52468169, -0.7983058 ],
       [ 0.11817452,  0.99915458],
       [-0.46282288, -0.35442231],
       [-0.04327923,  0.77653254],
       [-0.86201473, -0.46493419],
       [ 2.3737778 , -1.34068943],
       [ 0.38060072,  0.71505234],
       [ 0.06250958,  1.17600631]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  8
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  8  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
```

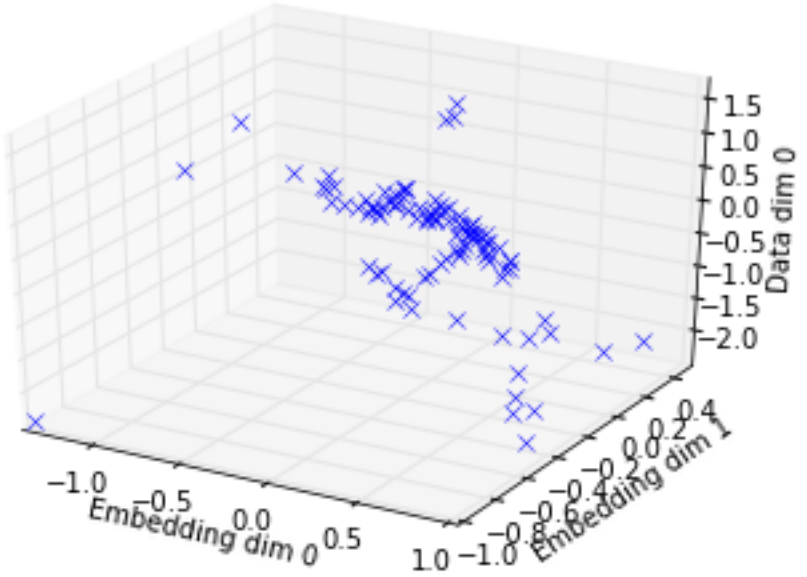
```
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[8] = -451.969854902

{'alpha': array([[ 0.93425698,  0.94669091]]), 'beta': array([[
1.1571807]]), 'Z': array([[-0.46076456, -0.48799155],
       [ 1.49238207,  0.42383834],
       [-0.52093387, -0.79439762],
       [ 0.11823292,  1.00480939],
       [-0.45772134, -0.36108454],
       [-0.04354249,  0.77208145],
       [-0.85351841, -0.46691729],
       [ 2.37360866, -1.34044319],
       [ 0.38045027,  0.71483524],
       [ 0.06277168,  1.1725672 ]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  9
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  9  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[9] = -441.732127347

{'alpha': array([[ 0.93317427,  0.94585764]]), 'beta': array([[
1.00088704]]), 'Z': array([[-0.45396449, -0.49101849],
       [ 1.49259589,  0.42397335],
       [-0.52247345, -0.79600307],
       [ 0.11820893,  1.00248644],
       [-0.45981702, -0.35834774],
       [-0.04343434,  0.77390993],
       [-0.85700863, -0.46610265],
       [ 2.37367814, -1.34054434],
       [ 0.38051208,  0.71492443],
       [ 0.06266401,  1.17397996]]), 'sf2': array([[
1.00000000e-10]])}
Iteration  10
Dispatching statistics Map-Reduce...
Done! statistics Map-Reduce took  9  seconds
Calculating global statistics...
Done! global statistics took  0  seconds
Dispatching embeddings Map-Reduce to run in background...
Waiting for embeddings Map-Reduce to finish...
Done! embeddings Map-Reduce took  1  seconds
F[10] = -437.090173637

final F=-437.090173637
```

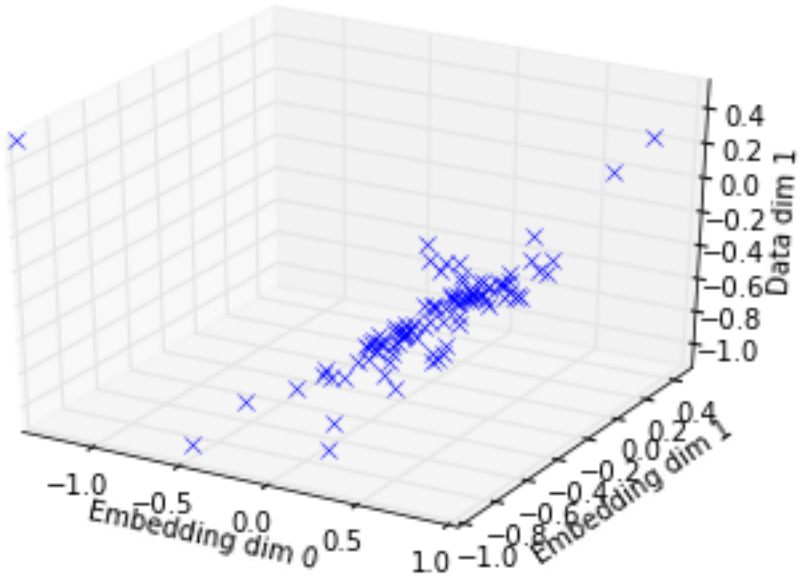
In [43]:
```
import show_embeddings
class empty:
    pass
```

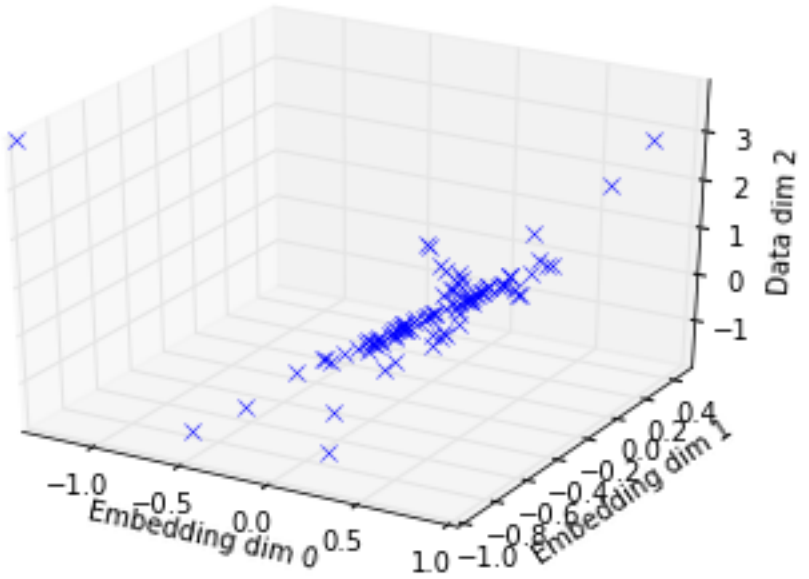
```
disp_opt = empty()
disp_opt.verbose = True
disp_opt.dimension = [0, 1]
disp_opt.output_dimension = [0, 1, 2]
disp_opt.plot2d = True
disp_opt.plot3d = False
args = ['./easydata/']
show_embeddings.run(disp_opt, args)
```

```
Displaying X in './easydata/'...
alpha: [ 0.93425698  0.94669091]
beta : 1.15718070139
sf2  : 1e-10
```




1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
Data dim 0
−1.0
−0.5
0.0
0.5
1.0
Embedding dim 0
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
Embedding dim 1


0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
Data dim 1
−1.0
−0.5
0.0
0.5
1.0
Embedding dim 0
−1.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
Embedding dim 1


3
2
1
0
−1
Data dim 2
−1.0
−0.5
0.0
0.5
1.0
Embedding dim 0
−1.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
Embedding dim 1

In []: