validation_runs

March 24, 2019

```
In [1]: import ingestor, modeller, fitter
        import numpy as np
        from matplotlib import pyplot as plt

In [47]: plt.rc('text', usetex=True)
         plt.rc('font', family='serif')

In [48]: from importlib import reload

         reload(ingestor)
         reload(modeller)
         reload(fitter)

Out[48]: <module 'fitter' from '/media/dwu402/Data/wrap-mad/fitter.py'>

In [3]: context = ingestor.initialise_context()
        ingestor.read_run_file(context, "runs/mouse4.3.run")

In [4]: model = modeller.Model(context)

In [5]: solver = fitter.Fitter()
        solver.construct_objectives(context, model)

In [6]: solver.construct_problems()

In [9]: for rhoi in logitspace(-2, 6, num=20):
            solver.solve(rhoi)

In [10]: solver.solutions

Out[10]: {'1000': [      fun: 0.25705175997649854
            hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
                 jac: array([ 0.03637033,  0.00942941,  0.00134705,  0.00362401, -0.01189578,
                  -0.00036214, -0.00244961,  0.00078433,  0.00581754])
             message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
                nfev: 77
                 nit: 30
              status: 0
             success: True
         x: array([0.        , 0.1403137 , 6.34427773, 0.        , 1.23773588,
        7.88697652, 5.23177699, 0.5933902 , 0.94568745])],
'0.01': [      fun: 0.024463866451051055
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-2.41444582e-05,  3.04777362e-05,  3.94982027e-07, -2.78702603e-05,
         7.99406411e-05,  2.06282584e-05, -9.19200566e-05, -1.54723281e-06,
         6.32921275e-05])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 21
       nit: 1
    status: 0
   success: True
         x: array([3.00000000e-01, 1.00000000e+00, 7.00000000e-01, 2.00000000e+00,
        1.00000000e+00, 1.00000000e+00, 1.00000000e+00, 1.00000000e+00,
        9.99999995e-04])],
'0.026366508987303583': [      fun: 0.02448206421246037
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 3.50774289e-06,  6.97212818e-06, -4.00951314e-06,  9.02378977e-08,
        -9.42940546e-07, -4.93753213e-06,  3.36432848e-06,  5.55137502e-06,
         2.47321448e-05])
   message: b'CONVERGENCE: NORM_OF_PROJECTED_GRADIENT_<=_PGTOL'
      nfev: 17
       nit: 12
    status: 0
   success: True
         x: array([0.01668625, 0.        , 2.19298475, 1.68940658, 0.80696864,
        6.61425893, 0.64500409, 2.10558408, 0.        ])],
'0.06951927961775606': [      fun: 0.024734582228442277
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 6.32593492e-07,  6.03855059e-06,  8.49931644e-06,  1.87244805e-07,
        -2.78295647e-06, -5.42471482e-07, -6.78797082e-06,  9.79253754e-06,
         1.00901722e-05])
   message: b'CONVERGENCE: NORM_OF_PROJECTED_GRADIENT_<=_PGTOL'
      nfev: 31
       nit: 24
    status: 0
   success: True
         x: array([5.74524052e-04, 0.00000000e+00, 2.13402223e+00, 9.44209484e+00,
        1.10500108e+00, 3.47873952e+01, 7.84619492e-01, 1.99428157e+00,
        0.00000000e+00])],
'0.18329807108324356': [      fun: 0.026695710813195973
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-3.43183186e-06,  4.82955668e-04,  4.65045991e-06, -2.22880034e-05,
         1.89476091e-05,  1.99134519e-05, -1.70720204e-05, -5.27306891e-06,
         1.46812481e-03])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 33
       nit: 23
    status: 0
   success: True
         x: array([0.        , 0.        , 1.9236622 , 3.55710068, 0.73594858,
        3.73079805, 1.61247417, 2.62836749, 0.        ])],
'0.4832930238571752': [      fun: 0.03235963654155967
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-0.00014097,  0.00035512, -0.00153483,  0.00030498, -0.00082758,
        -0.00174051,  0.0011131 ,  0.00078654,  0.00649614])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 38
       nit: 24
    status: 0
   success: True
         x: array([0.        , 0.        , 1.15669107, 3.41454271, 0.90047206,
        2.01902722, 1.67196674, 4.12028325, 0.        ])],
'1.2742749857031335': [      fun: 0.03134266760161627
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-1.21075399e-05, -6.21704256e-06, -4.01003557e-05, -1.65941512e-06,
         4.16108098e-05,  1.28460049e-06,  2.14933073e-05, -1.13097380e-05,
         1.04198736e-03])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 197
       nit: 117
    status: 0
   success: True
         x: array([12.67673425, 99.99416089,  0.57351643,  8.29852144,  0.76587672,
        46.16372515,  7.19413601,  5.82425306,  0.        ])],
'3.359818286283781': [      fun: 0.06998652531134526
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-2.17693404e-03, -3.79941140e-03,  6.75928097e-03, -8.17986832e-04,
         1.29487698e-02,  9.72272885e-06,  1.97767880e-02, -7.62823255e-03,
        -2.51849050e-03])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 92
       nit: 15
    status: 0
   success: True
         x: array([0.        , 7.84613041, 0.        , 1.31385123, 2.24358974,
        9.10552582, 1.04961366, 4.24569061, 0.        ])],
'8.858667904100823': [      fun: 0.11327255991650063
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 7.72386236e-04,  2.28229076e-03,  6.82498433e-04, -2.87950582e-06,
         3.91082900e-05,  1.28376980e-05, -9.75059628e-05,  7.69729237e-05,
         1.81838137e-02])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 77
       nit: 52
    status: 0
   success: True
         x: array([ 0.        ,  0.        ,  0.        , 10.48332092,  1.6714006 ,
         9.8632435 ,  4.2176784 ,  2.81152605,  0.        ])],
'23.357214690901213': [      fun: 0.20320559362937984
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 0.01351528, -0.00558197,  0.00199296,  0.0018588 , -0.01127916,
        -0.00234596,  0.00583262, -0.00246992, -0.00461297])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 103
       nit: 26
    status: 0
   success: True
         x: array([0.42750788, 0.        , 2.04953218, 0.        , 1.31012753,
        4.48236175, 1.82316951, 1.89485669, 0.24518563])],
'61.584821106602604': [      fun: 0.41987979509713114
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-0.02674027, -0.01497678, -0.00141475,  0.08106391, -0.0965789 ,
        -0.04832687,  0.01644304,  0.06155291,  0.0284845 ])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 14
       nit: 4
    status: 0
   success: True
         x: array([0.88381629, 0.36275839, 1.41016007, 1.83595136, 1.08836533,
        1.96266307, 1.49936886, 0.19430534, 0.48026189])],
'162.3776739188721': [      fun: 0.2787486008220703
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 2.83997566e-03, -3.55434801e-03, -4.69548225e-04,  2.78081064e-02,
        -6.15448573e-02, -2.08524985e-03,  1.01880713e-03,  9.72264687e-05,
         2.27872842e-02])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 49
       nit: 19
    status: 0
   success: True
         x: array([5.08439288, 2.34247678, 8.20556048, 1.05153184, 0.63773864,
        8.23897126, 9.73152164, 0.        , 0.71156459])],
'428.13323987193957': [      fun: 0.28132500309417097
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 0.02245196,  0.02176788, -0.00369698, -0.02933825,  0.04559425,
        -0.00963221,  0.00244634,  0.00140488,  0.00410286])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 84
       nit: 17
    status: 0
   success: True
         x: array([0.22670612, 0.45987833, 2.13809557, 0.82371328, 0.75621601,
        2.65582161, 2.53173744, 0.9773532 , 0.83742272])],
'1128.8378916846884': [      fun: 0.2943613425739074
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ 0.00037693, -0.00672576, -0.00140636, -0.01538797,  0.03648566,
        -0.00400027,  0.00493511, -0.00066497,  0.00922127])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 63
       nit: 24
    status: 0
   success: True
         x: array([0.40265268, 0.03620073, 3.7549415 , 0.03256845, 0.22860387,
        4.74487525, 4.00886714, 1.03495193, 1.03795394])],
'2976.3514416313133': [      fun: 3.7139373441123933
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ -0.78443828,   0.86493576,  -1.21851761,  15.72189807,
        -29.53151538,  -8.55170876,   6.42891733,   0.20059324,
         -3.73595014])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 30
       nit: 1
    status: 0
   success: True
         x: array([3.e-01, 1.e+00, 7.e-01, 2.e+00, 1.e+00, 1.e+00, 1.e+00, 1.e+00,
        1.e-03])],
'7847.5997035146065': [      fun: 0.2882203385315192
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-0.021215  ,  0.00248915,  0.00180936, -0.00013196, -0.00644569,
         0.00027293,  0.00371364, -0.00020568,  0.00145821])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 122
       nit: 44
    status: 0
   success: True
         x: array([ 0.79661611,  3.40276971,  6.72827574, 25.87402871,  1.49132616,
        16.11654432,  1.08625064, 13.94794233,  3.84265172])],
'20691.3808111479': [      fun: 1.3180323071592595
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-1.11897029e-02,  3.56681660e-02, -6.59804205e-02,  4.51225096e-01,
        -2.44579050e-01,  2.74210268e+00, -2.47760689e+00, -1.45903890e-03,
         1.11168163e+00])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 95
       nit: 15
    status: 0
   success: True
         x: array([0.70432246, 0.43071838, 1.03793708, 1.5350546 , 0.8584425 ,
        0.92074355, 1.17730864, 1.06278805, 0.79738531])],
'54555.947811685146': [      fun: 5.6030792160528335
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
```

```
       jac: array([ -0.91628539,   1.01512711,  -0.62742524,  36.16320333,
        -72.67153882, -38.71244551,  38.37058285,   0.25537551,
        -11.97487846])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 27
       nit: 1
    status: 0
   success: True
         x: array([3.e-01, 1.e+00, 7.e-01, 2.e+00, 1.e+00, 1.e+00, 1.e+00, 1.e+00,
        1.e-03])],
'143844.988828766': [      fun: 4.3268778243805475
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-13.85480213,   5.95969107,  -5.98973991,  -2.22445764,
          8.49719345,  -3.05790926,   6.45392416,  -1.96842477,
         -0.83913776])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 54
       nit: 3
    status: 0
   success: True
         x: array([0.36559834, 0.99252647, 0.73660456, 1.99211504, 1.0870829 ,
        1.07619343, 0.91717268, 1.04912149, 0.03680923])],
'379269.0190732246': [      fun: 6.145581701667144
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([ -0.49510609,   0.53594205,  -0.27013529,  29.67405092,
        -57.50908749, -28.14730569,  27.45181931,   0.11572738,
        -10.42719581])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 81
       nit: 1
    status: 0
   success: True
         x: array([3.e-01, 1.e+00, 7.e-01, 2.e+00, 1.e+00, 1.e+00, 1.e+00, 1.e+00,
        1.e-03])],
'1000000.0': [      fun: 4.668268570326693
  hess_inv: <9x9 LbfgsInvHessProduct with dtype=float64>
       jac: array([-21.96385611,  10.00974427, -10.87117517,  -4.78763771,
         17.32519211,  -3.70358529,   7.22817311,  -1.87979359,
         -0.98696143])
   message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
      nfev: 66
       nit: 3
    status: 0
   success: True
         x: array([0.37507691, 0.98761736, 0.72970607, 1.99308578, 1.08593965,
        1.07024594, 0.92228612, 1.0464066 , 0.03333743])]}
```

0.1 Validation

```
In [52]: ## Validation of the outer jacobian
         ca = fitter.ca
         dHdc = ca.hcat([ca.gradient(solver._inner_objective._obj_1, ci) for ci in model.cs]).re
         d2Jdc2 = ca.hcat([ca.jacobian(solver._inner_objective.inner_jacobian, ci) for ci in mod
         dJ2dcdp = ca.hcat([ca.jacobian(solver._inner_objective.inner_jacobian, pi) for pi in mo
```

```
In [53]: dHdc_fn = ca.Function("dhdcfn", solver._inner_objective.input_list, [dHdc])
         d2Jdc2_fn = ca.Function("d2jdc2", solver._inner_objective.input_list, [d2Jdc2])
         dJ2dcdp_fn = ca.Function("d2jdcdp", solver._inner_objective.input_list, [dJ2dcdp])
```

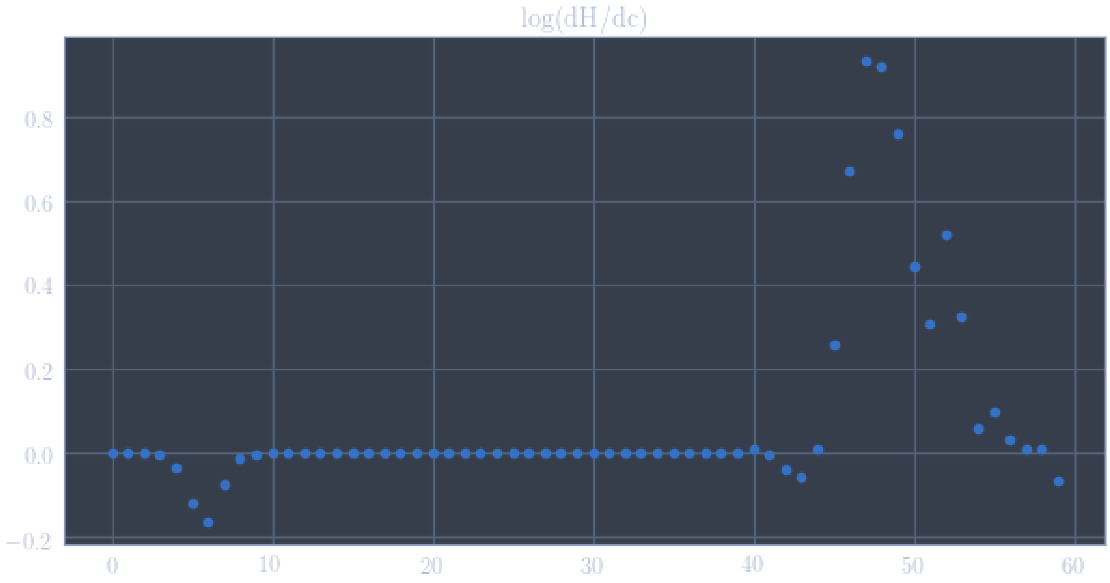
```
In [56]: in_arg = [model.observation_times, *fitter.argsplit(solver.problems[0].cache.recent, 3)
```

```
In [57]: dhdc_eval = dHdc_fn(*in_arg)
         d2jdc2_eval = d2Jdc2_fn(*in_arg)
         dj2dcdp_eval = dJ2dcdp_fn(*in_arg)
```

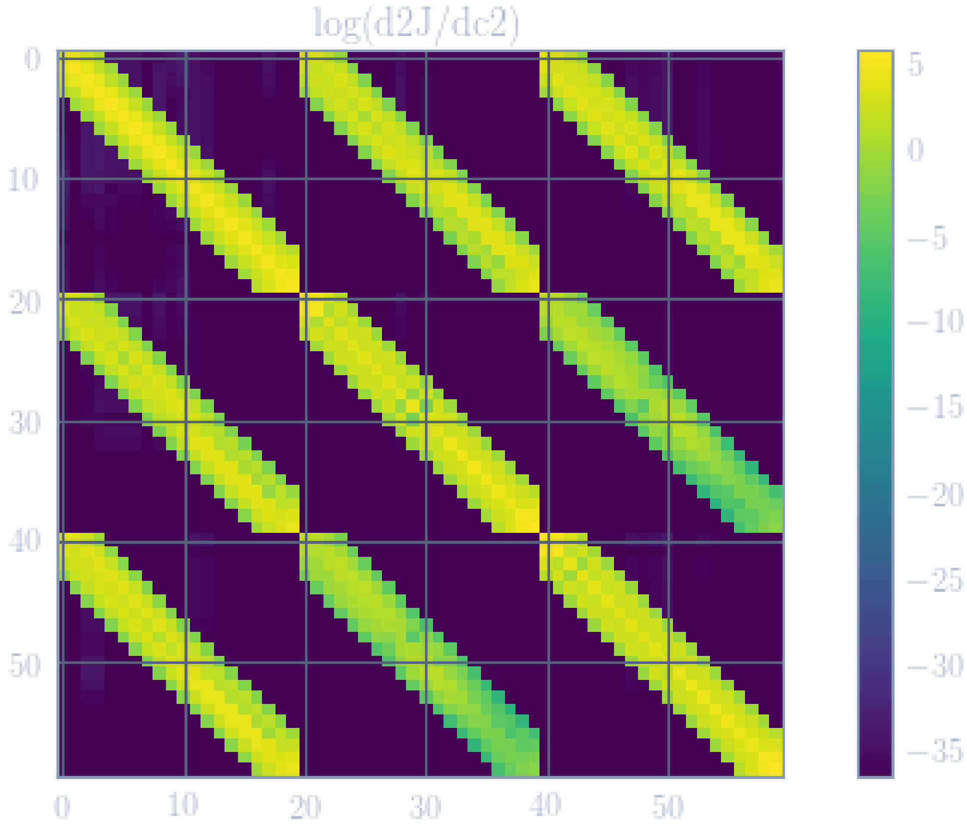
```
In [58]: import numpy as np
         def numerical_log(matrix):
             return np.log(np.fabs(np.array(matrix))+1e-16)

         import matplotlib.pyplot as plt
         plt.rcParams['figure.figsize'] = [10, 5]
```

```
In [59]: plt.plot(np.array(dhdc_eval).reshape(-1,), 'o')
         plt.title("log(dH/dc)")
```

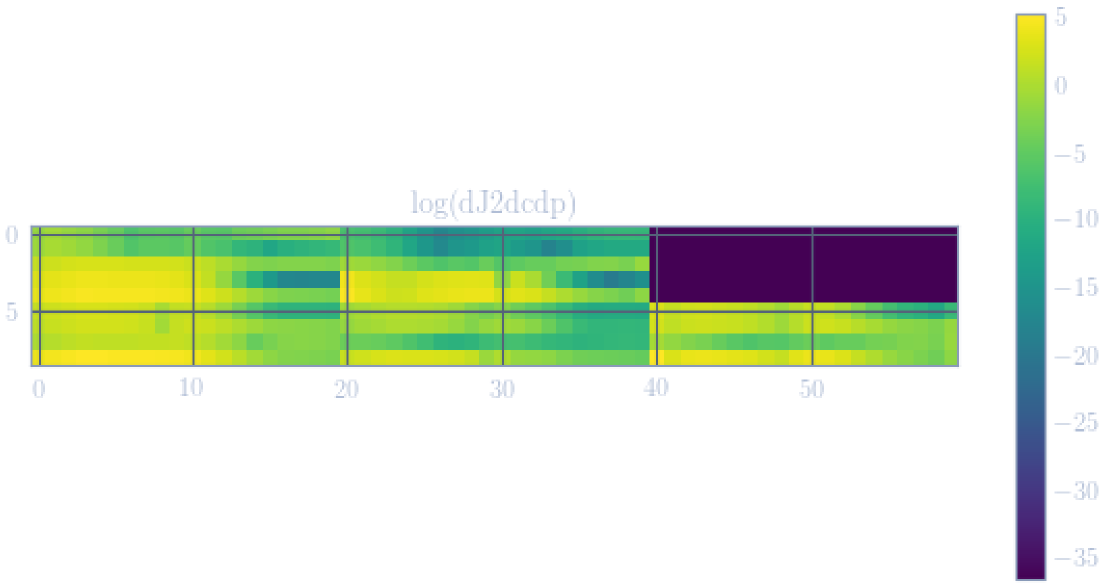
```
Out[59]: Text(0.5, 1.0, 'log(dH/dc)')
```



```
In [60]: plt.imshow(numerical_log(d2jdc2_eval))
         plt.colorbar()
         plt.title("log(d2J/dc2)")

Out[60]: Text(0.5, 1.0, 'log(d2J/dc2)')
```

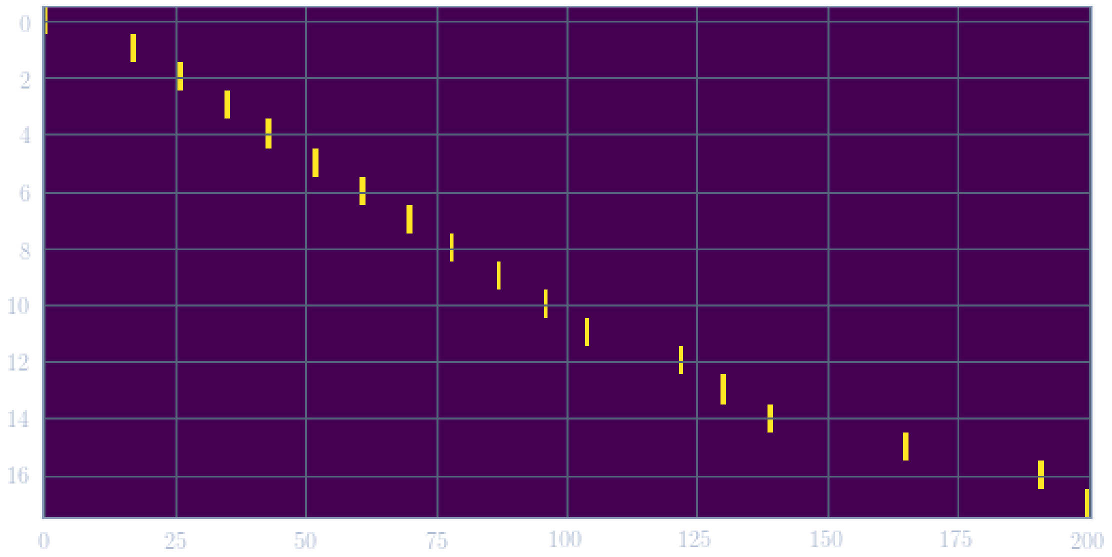


```
In [61]: plt.imshow(numerical_log(dj2dcdp_eval).T)
         plt.colorbar()
         plt.title("log(dJ2dcdp)")

Out[61]: Text(0.5, 1.0, 'log(dJ2dcdp)')
```



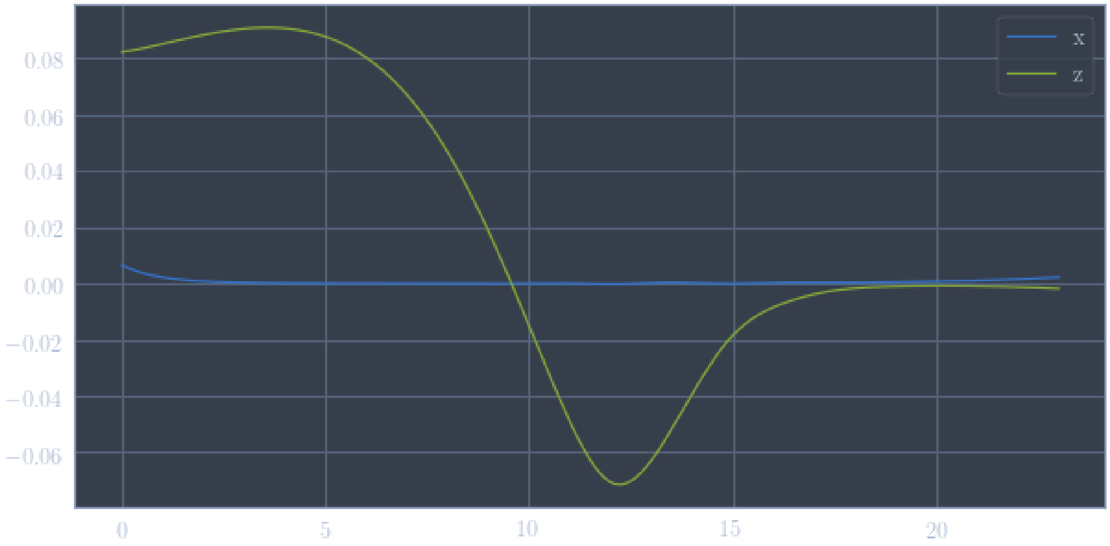
```
In [62]: H_num = solver._inner_objective.generate_collocation_matrix(context['datasets'][0], mod
         plt.imshow(H_num, aspect='auto')
```

```
Out[62]: <matplotlib.image.AxesImage at 0x7fe1bc06fb70>
```



```
In [63]: # create and profile calls

         obj_fn, obj_jac = solver._inner_objective.create_objective_functions(model, context['da
         c_test = np.array(solver.problems[0].cache.recent)

         %timeit obj_fn(c_test, [0.3, 1, 0.7, 2, 1, 1, 1,1 , 1e-4], rho=1000)

33.8 ms ś 237 ţs per loop (mean ś std. dev. of 7 runs, 10 loops each)
```

```
In [64]: getx = ca.Function("getx", [model.ts, *model.cs], model.xs)

In [65]: xs = np.array([np.array(i) for i in getx(model.observation_times, *fitter.argsplit(solv
         plt.plot(model.observation_times, np.hstack([xs[0], xs[2]]))
         plt.legend("xz")

         iv = [xs[i][0].item() for i in range(3)]
```

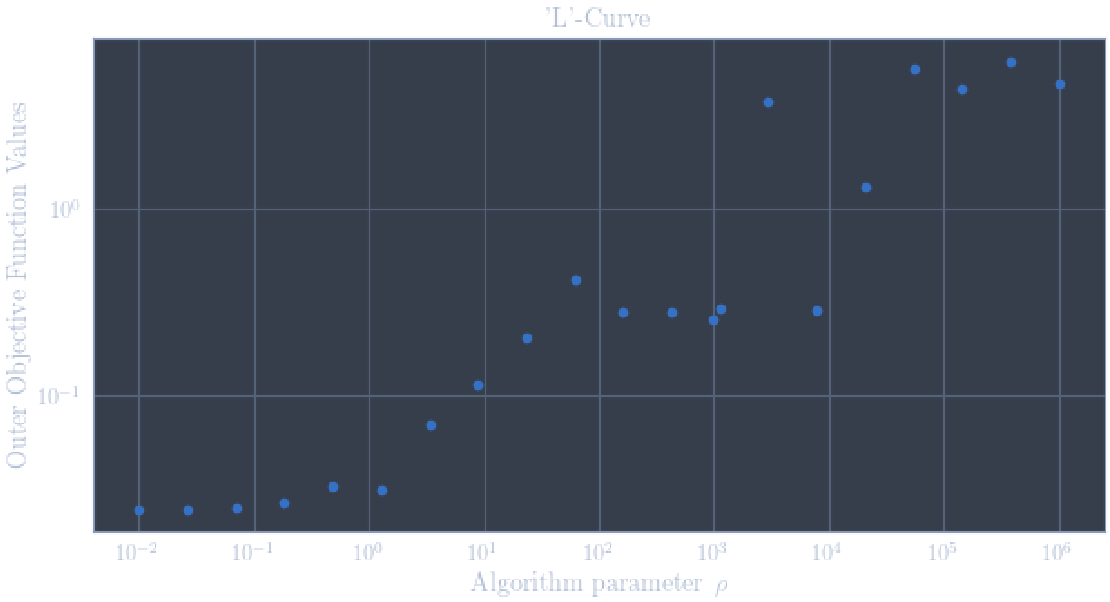


```
In [66]: from scipy import integrate

         sol = integrate.solve_ivp(lambda t, y: context['model'](t, y, [0.92255063, 0.55098101,
                  2.21360157, 2.85341497, 1.13601111, 0.55253756]), [0, 24], iv)

         sol.y[2]

Out[66]: array([ 8.23747200e-02,  2.43885910e-02, -4.67347239e-01, -5.68521513e-01,
                -4.07164954e-01, -2.31045165e-01, -1.37203279e-01, -1.03233478e-01,
                -1.11901414e-01, -1.74140638e-01, -4.10553674e-01, -1.12822415e+00,
                -6.09662589e+00, -4.73593268e+01, -5.09094718e+02, -4.65586865e+03,
                -2.88217278e+04, -1.26081360e+05, -5.48574606e+05, -2.92729335e+06,
                -1.81766342e+07, -2.00298078e+07])
```

```
In [67]: outer_evals = {r:v[0].fun for r, v in solver.solutions.items()}

         outer_list = np.array([[float(key), value] for key, value in outer_evals.items()])
```

```
In [68]: plt.loglog(*outer_list.T, 'o')
         plt.xlabel(r"Algorithm parameter $\rho$ ")
         plt.ylabel(r"Outer Objective Function Values")
         plt.title("'L'-Curve")
```

```
Out[68]: Text(0.5, 1.0, "'L'-Curve")
```

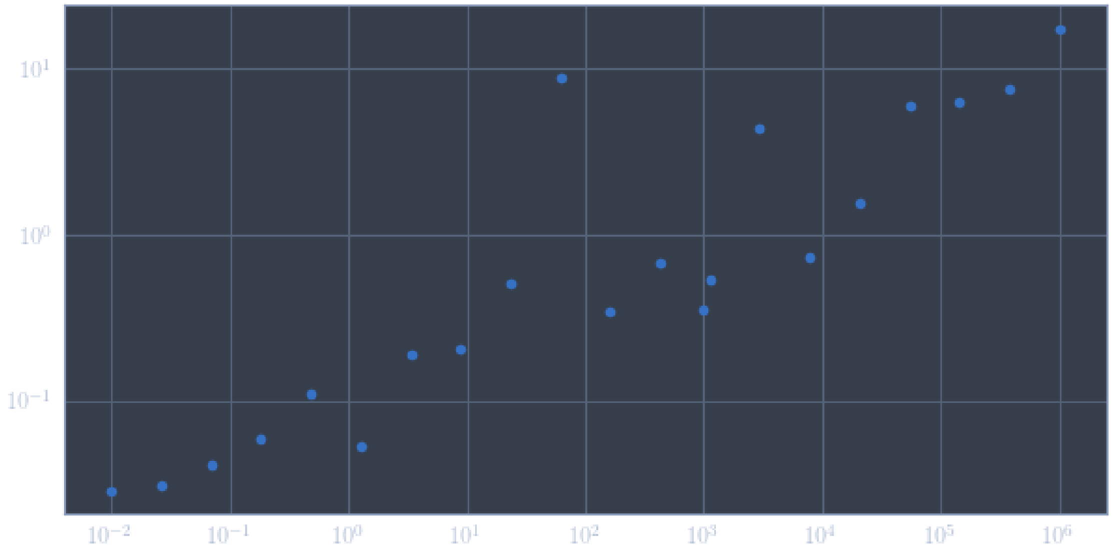


```
In [69]: def gen_key(sol_key, solution):
             return "y".join(map(str,solution.x)) + "r" + sol_key
```

```
In [70]: inner_evals = dict()
         for soli in solver.solutions:
             key = gen_key(soli, solver.solutions[soli][0])
             inner_evals[soli] = solver.problems[0].cache.results[key]
```

```
In [71]: new_list = []
         for key in inner_evals.keys():
             new_list.append((float(key), inner_evals[key].fun))
         new_list = np.array(new_list)
         plt.loglog(*np.array(new_list).T, 'o')
```

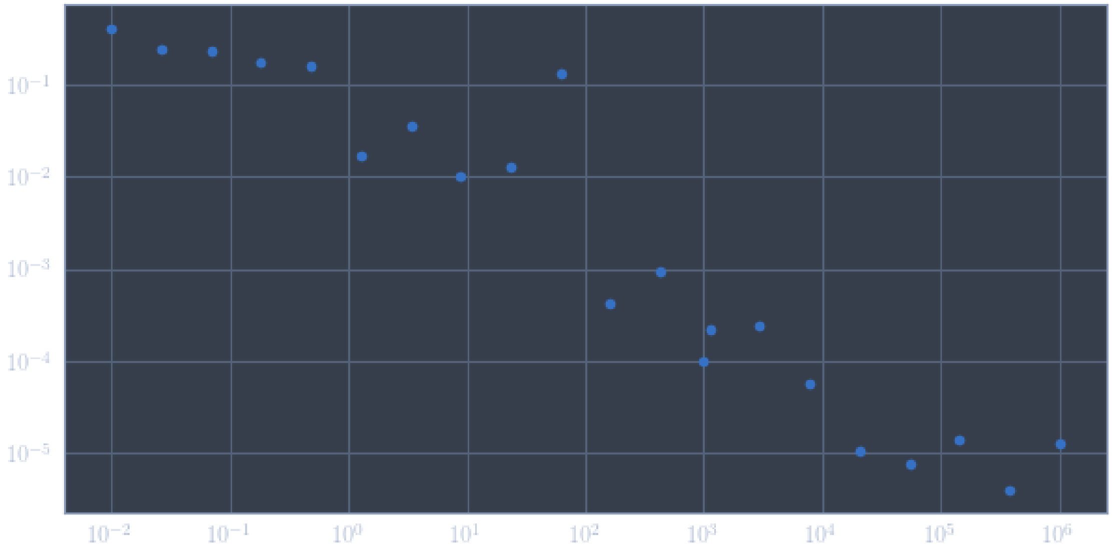
```
Out[71]: [<matplotlib.lines.Line2D at 0x7fe1b5aa10f0>]
```



```
In [72]: diff_field_value = [[okey, (ivalue-ovalue)/(ikey)] for (ikey, ivalue), (okey, ovalue) i
```

```
In [73]: plt.loglog(*np.array(diff_field_value).T, 'o')
```

```
Out[73]: [<matplotlib.lines.Line2D at 0x7fe1b58dff60>]
```



```
In [ ]:
```