README for qORAC matlab files


```
Recreates the images in
"Maintaining maximal metabolic flux by gene expression control" by
Planque, Hulshof, Teusink and Bruggeman, Cell Systems.

The code is available at
http://www.few.vu.nl/~rplanque/Research/qORAC/
```

First, within the directory in which you save the matlab files, create two directories, "pics" and "Symbolic_functions". The first contains output figures, the second contains the matlab symbolic functions with which the programs do most computations. These functions are specific for each pathway, and need to be generated each time parameters, kinetics or pathway structure are changed.

Note that the code files are not generic code for arbitrary pathways. Each file specifies the qORAC framework for a specific pathway. In future releases we plan to make this code more generic, so that it is easier to play around with different pathways.

Figure 2

To recreate the enzyme synthesis relations in **Figure 2B** for the network shown in Figure 2A, run

```
>> sensor_optimum_enz_CN_newton
```

The corresponding image is `CN_1_input_output.jpg`

To recreate the subplots in **Figure 2C**, run

```
>> [t,x,x0] = daes_CN(1,1,1,50);
```

This gives several figures, called `CN_1_....jpg`.

To rerun the script without computing the symbolic functions again, do

```
>> [t,x,x0] = daes_CN(1,1,0,50);
```

Figure 3

To recreate **Figure 3A**, use

```
>> daes_extra_param
```

This example uses metabolites 2, 3 and 6 as sensors.

For **Figure 3B**, we choose different sensors (namely 2, 4 and 6). These do follow the optimum (so the sensors are valid), but for which the combined pathway + qORAC does not steer to optimum, do

```
>> daes_extra_param_wrong
```

Figure S1

For the complicated double branched pathway, with two allosteric interactions, Figure S1 in the Supplement, run

```
[t,x,x0] = daes_double_branched_sym;
```

Figure S2

This is the minimal ICs plot in the Supplement.

```
>> [t,x,v] = daes_CN_minimal_ICs(1,1,0,50);
```

Figure S3

The code to make Supplementary Figure S3 is

```
>> daes_linearchain_reversal
```

You may have to run the program more than once to calculate through the critical thermodynamic equilibrium point (despite the fact that the equations have been changed appropriately to make the Implicit Function Theorem valid in this point). In each new run, a different initial condition is chosen. A good run has time running from 0 to 300. After running the code once, you can run it faster by not computing the symbolic functions again, using

```
>> daes_linearchain_reversal(1)
```