Installing

Microbial communities are essential for global ecosystems and human health. Computational modeling of microbial consortia is thus a major goal in systems biology and microbial ecology. BacArena is a project to simulate bacterial behaviour in communities. A lot of progress is done in the last years to gain genome wide metabolic reconstructions of certain organisms, which open a wide field of mathematical analysis. One of this new methods is fux balanced analysis (fba) to estimate optimal metabolic fluxes under certain constraints. With this work, advanced models are build which are available in an exchangeable format (SBML). Some of these models can be found in[http://bigg.ucsd.edu/]. The idea of this project is to use this existing reconstructions and put them in a spatial and temporal environment to study their possible interactions. This is achieved by the combination of agent based modeling with fba. Each bacterium is considered as an agent with individual states, own properties and rules to act. Agents are located on a grid where they can move and interact via metabolic exchanges computed by fba. The starting point for our project is curiosity of what could be done with this huge models. We just throw those models into an arena to see what kind of actions will evolve.

Installing

BacArena is avaible as a R package. The developmental version can be installed from GitHub with the devtools package:

```
library("devtools")

## Warning: package 'devtools' was built under R version 3.1.2

install_github("euba/BacArena", ref="rpkg")

## Downloading github repo euba/BacArena@rpkg
## Installing BacArena
## "C:/PROGRA~1/R/R-31~1.0/bin/x64/R" --vanilla CMD INSTALL  \
##   "C:/Users/eugen.bauer/AppData/Local/Temp/Rtmp0GErA6/devtools1cc4433d5cca/euba-BacArena-
##   --library="C:/Users/eugen.bauer/Documents/R/win-library/3.1"  \
##   --install-tests
```

Getting started

First we have to load the installed package in the workspace with

```
library("BacArena")
```

```
## Loading required package: sybil
## Loading required package: Matrix
## Loading required package: lattice
## Loading required package: RcppArmadillo

## Warning: package 'RcppArmadillo' was built under R version 3.1.2

## Loading required package: glpkAPI
## using GLPK version 4.47
```

Next we set-up the *Escherichia coli* core metabolic model

```
ecore <- model
```

The model is integrated in BacArena by default. Alternatively you can also load your own model of interest with commands of the sybil package and libSBML. After we loaded the model, we convert it into an object of class Bac by calling the constructor

```
bac <- Bac(ecore,deathrate=0.1,duplirate=1,
           growthlimit=0.05,growtype="exponential")
```

Now we have to set up an environment in which the organisms can interact

```
arena <- Arena(100,100)
```

Here, we chose 100 times 100 as the grid size of the environment. Next we want to put our created organism in its environment by

```
addOrg(arena,bac,amount=1,x=50,y=50)
```

With this command we added one individual of our bacterium in the middle of the environment (by its x and y position). Next we can add the substances to the environment

```
addSubs(arena,40)
```

Now we added all possible substances to the environment arena with a concentration 40 mmol per gridcell. Finally we can start the simulation with

```
eval <- simEnv(arena,20)
```

```
## iter: 1 bacs: 1
##      user  system elapsed
##      0.03    0.00    0.03
##      user  system elapsed
##      0.13    0.00    0.13
## iter: 2 bacs: 2
##      user  system elapsed
##      0.01    0.00    0.01
##      user  system elapsed
##      0.13    0.00    0.12
## iter: 3 bacs: 4
##      user  system elapsed
##      0.05    0.00    0.04
##      user  system elapsed
##      0.11    0.00    0.10
## iter: 4 bacs: 8
##      user  system elapsed
##      0.06    0.00    0.07
##      user  system elapsed
##      0.12    0.00    0.12
## iter: 5 bacs: 16
##      user  system elapsed
##      0.12    0.00    0.12
##      user  system elapsed
##      0.20    0.00    0.21
## iter: 6 bacs: 16
##      user  system elapsed
##      0.15    0.00    0.15
##      user  system elapsed
##      0.11    0.00    0.11
## iter: 7 bacs: 32
##      user  system elapsed
##      0.23    0.00    0.24
##      user  system elapsed
##      0.11    0.00    0.11
## iter: 8 bacs: 32
##      user  system elapsed
##      0.29    0.00    0.29
##      user  system elapsed
##      0.12    0.00    0.12
```

```
## iter: 9 bacs: 64
##    user  system elapsed
##    0.46    0.00    0.45
##    user  system elapsed
##    0.13    0.00    0.13
## iter: 10 bacs: 118
##    user  system elapsed
##    0.66    0.00    0.65
##    user  system elapsed
##    0.14    0.00    0.14
## iter: 11 bacs: 191
##    user  system elapsed
##    1.06    0.00    1.06
##    user  system elapsed
##    0.14    0.00    0.14
## iter: 12 bacs: 196
##    user  system elapsed
##    1.12    0.00    1.12
##    user  system elapsed
##    0.14    0.00    0.14
## iter: 13 bacs: 311
##    user  system elapsed
##    1.24    0.00    1.23
##    user  system elapsed
##    0.16    0.00    0.16
## iter: 14 bacs: 418
##    user  system elapsed
##    1.86    0.00    1.87
##    user  system elapsed
##    0.18    0.00    0.17
## iter: 15 bacs: 430
##    user  system elapsed
##    1.84    0.00    1.84
##    user  system elapsed
##    0.17    0.00    0.17
## iter: 16 bacs: 596
##    user  system elapsed
##    2.13    0.00    2.14
##    user  system elapsed
##    0.19    0.00    0.19
## iter: 17 bacs: 751
##    user  system elapsed
##    2.37    0.00    2.37
##    user  system elapsed
##     0.2     0.0     0.2
## iter: 18 bacs: 892
```

```
##    user  system elapsed
##    3.06    0.00    3.06
##    user  system elapsed
##    0.22    0.00    0.22
## iter: 19 bacs: 926
##    user  system elapsed
##    3.13    0.00    3.13
##    user  system elapsed
##    0.22    0.00    0.21
## iter: 20 bacs: 1123
##    user  system elapsed
##    3.66    0.00    3.67
##    user  system elapsed
##    0.26    0.00    0.27
```

The object eval stores all 20 simulation steps, that we performed. After we retrieve the eval object we can plot now the results of the simulation

```
plotCurves(eval)
```

Population

Substance concentrations
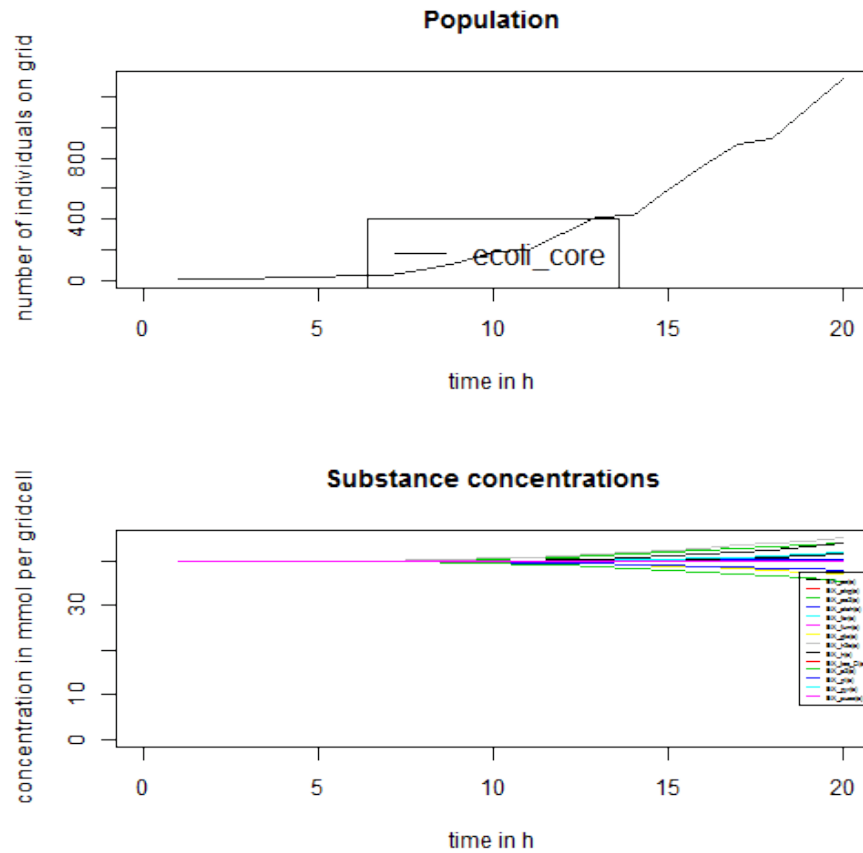
This will plot the growth curve and curves of substance concentration changes over the 20 simulation steps. If we are interested in the spatial and temporal changes of our constructed population we can use

```
evalArena(eval)
```

Population

..

Population
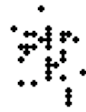
.:.

Population

⠞⠆

Population

Population

Population

Population

Population

Population

Population

Population

Population
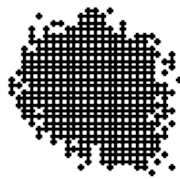
Population
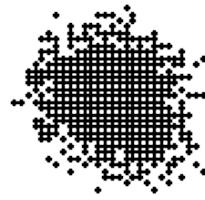
Population
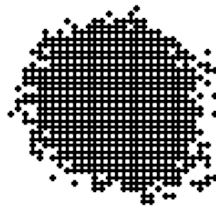
Population
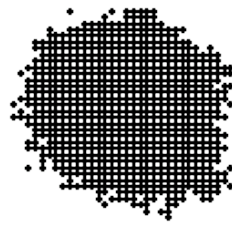
Population
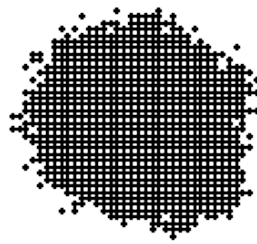
Population
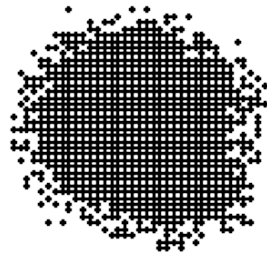
Population
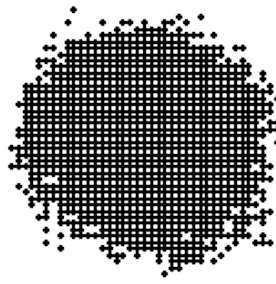
Population
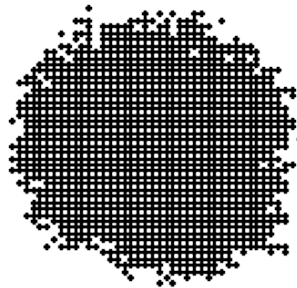
Population

This will produce multiple plots one by one for each simulation step with the spatial structure of the population (black dots represent individuals). We can also investigate the spatial change of the population together with the main subtrate glucose

```
evalArena(eval,c("population","EX_glc(e)"))
```

EX_glc(e)
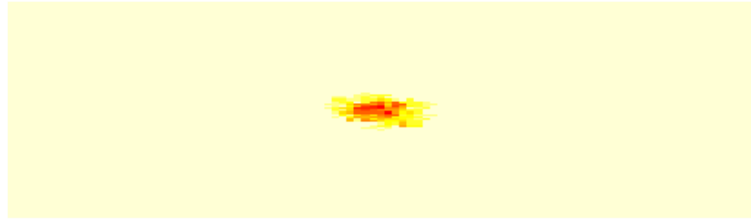
Population
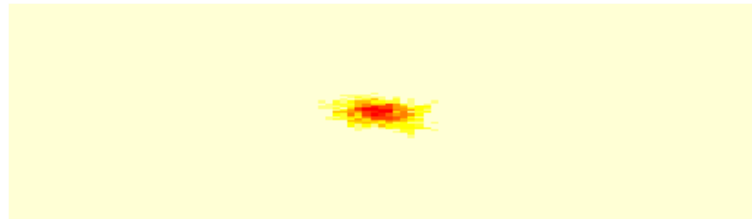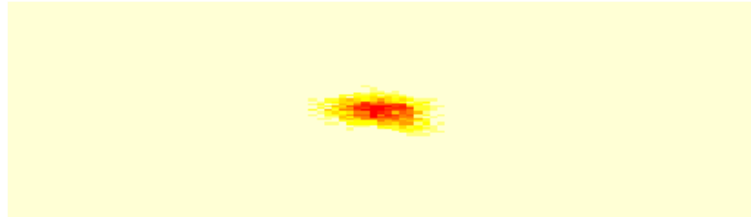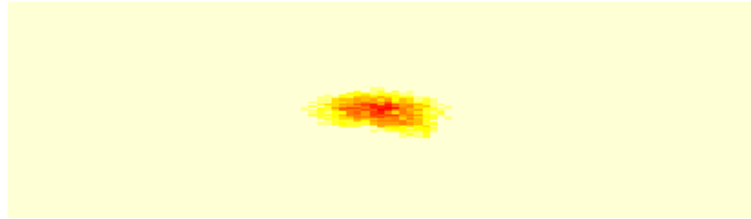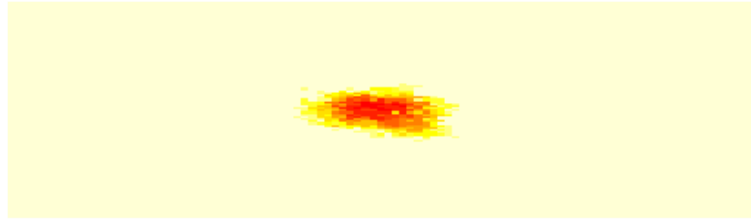
EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)
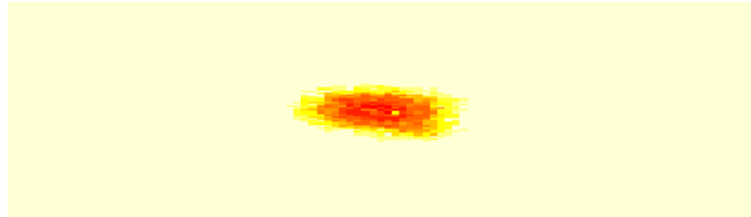
Population

EX_glc(e)
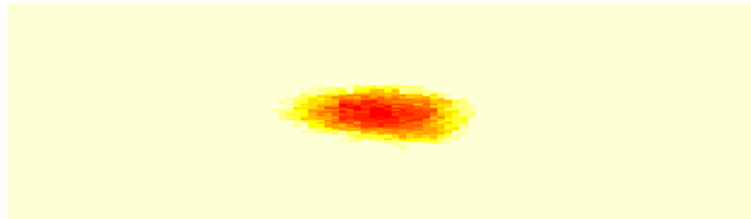
Population

EX_glc(e)
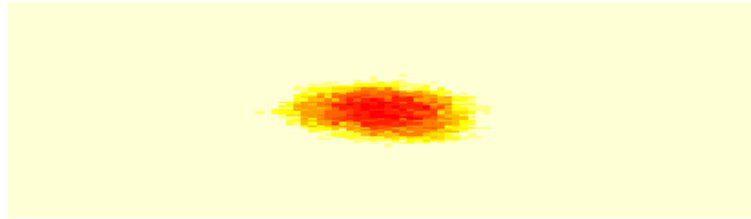
Population

EX_glc(e)
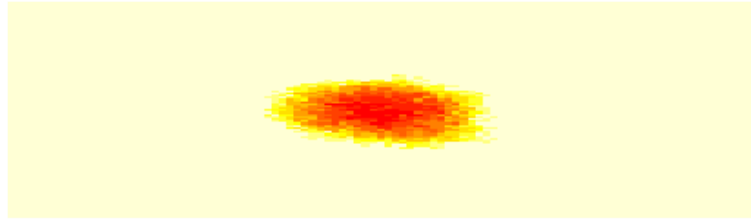
Population

EX_glc(e)
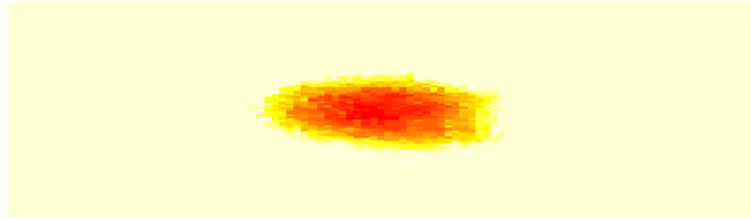
Population

EX_glc(e)
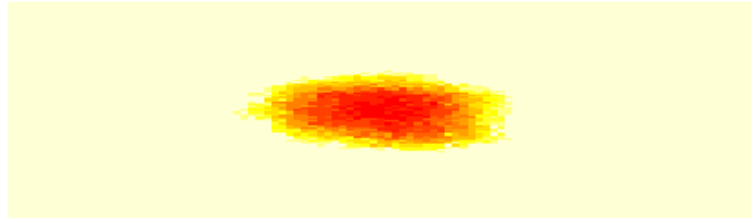
Population

EX_glc(e)
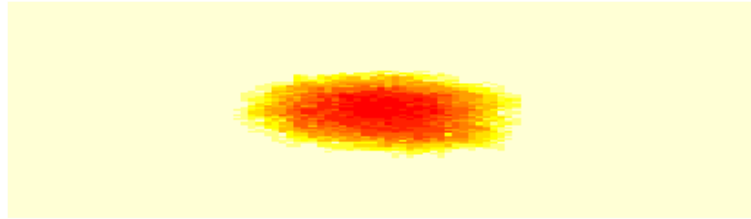
Population

EX_glc(e)
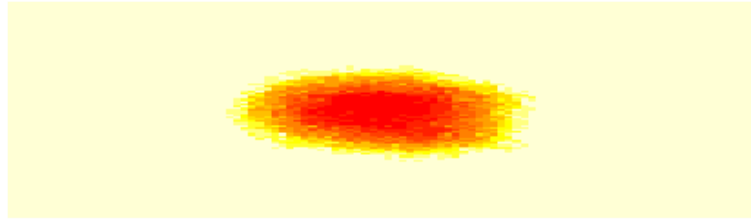
Population

EX_glc(e)

Population

At the same time we can also integrate the visualization of different phenotypes into the population

```
evalArena(eval,c("population","EX_glc(e)"),phencol=T)
```

EX_glc(e)
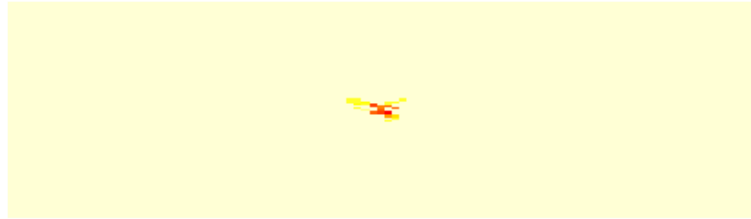
Population
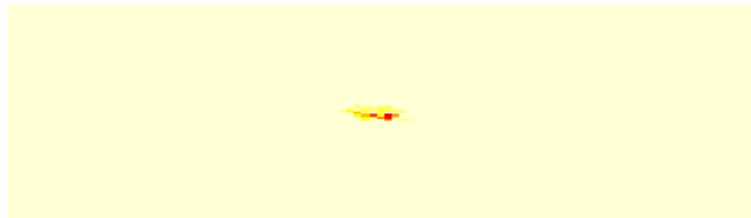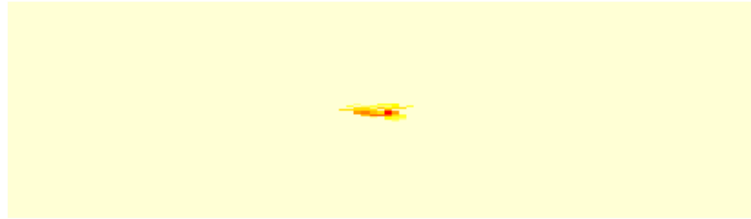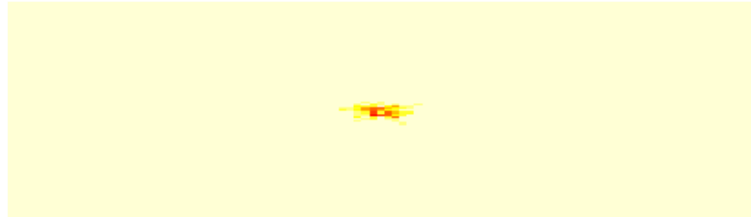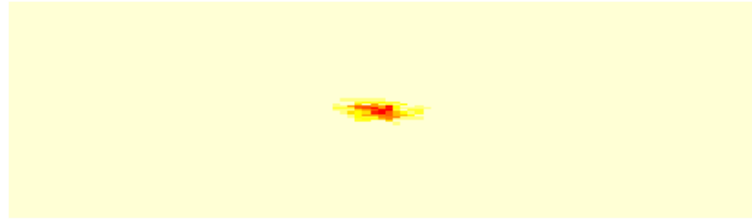
EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)

Population

EX_glc(e)
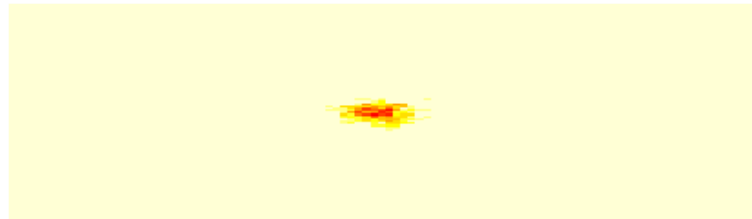
Population

EX_glc(e)
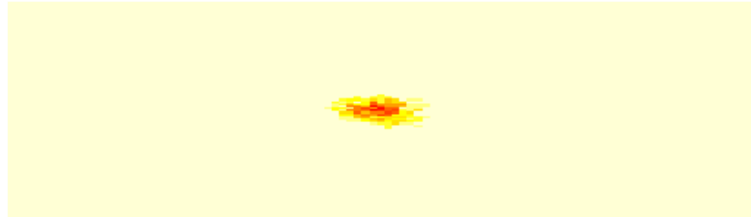
Population

EX_glc(e)
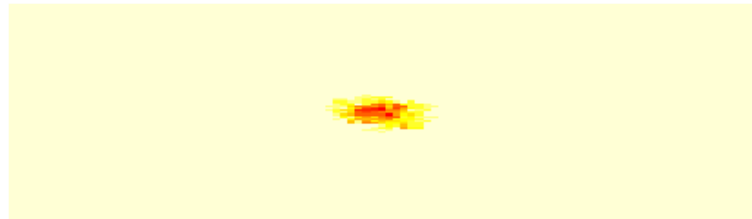
Population

EX_glc(e)
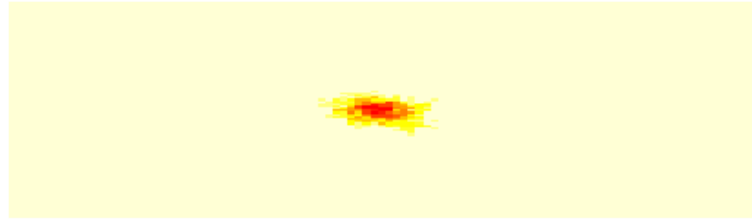
Population

EX_glc(e)
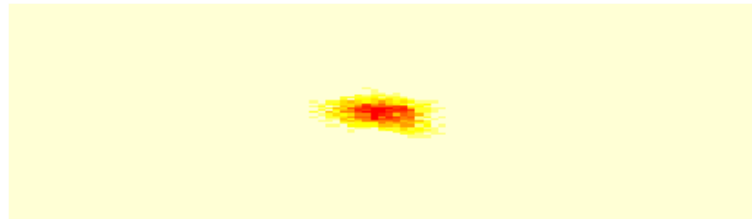
Population

EX_glc(e)
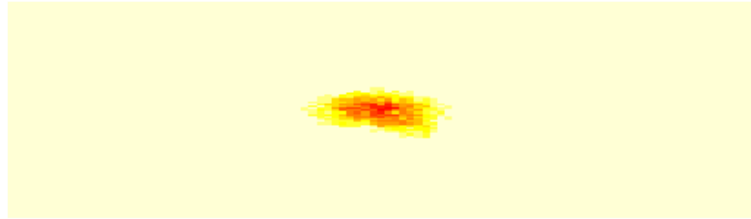
Population

EX_glc(e)
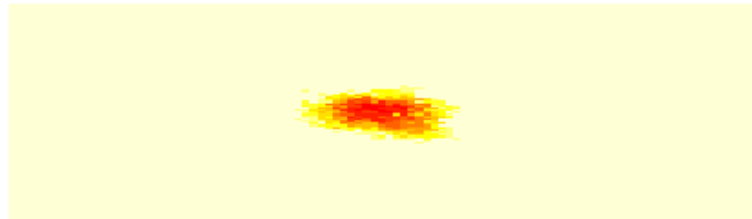
Population
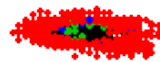
EX_glc(e)
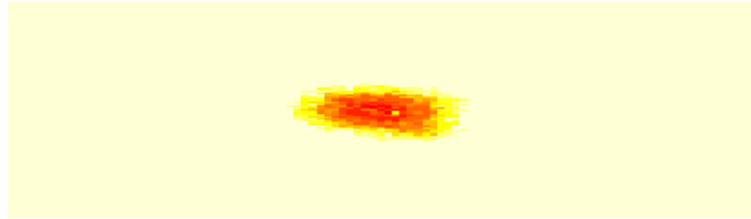
Population
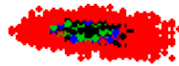
EX_glc(e)
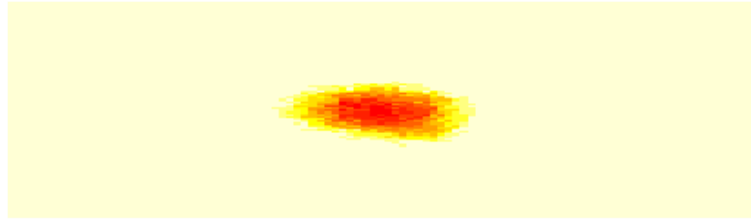
Population

EX_glc(e)
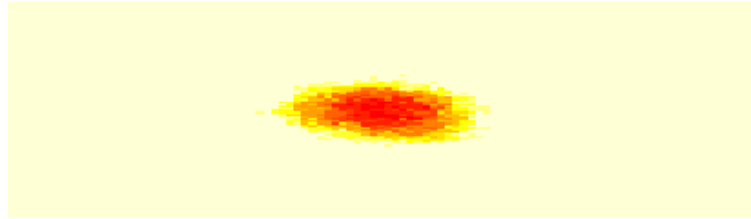
Population
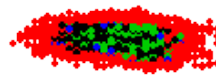
EX_glc(e)
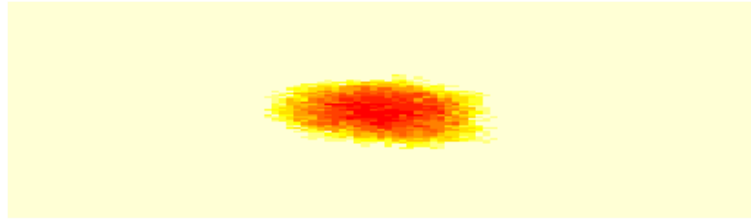
Population
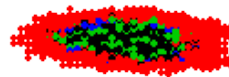
EX_glc(e)
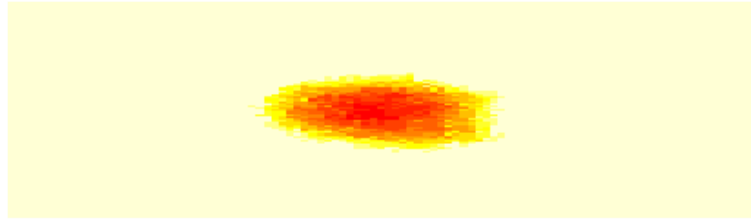
Population
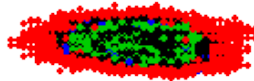
EX_glc(e)
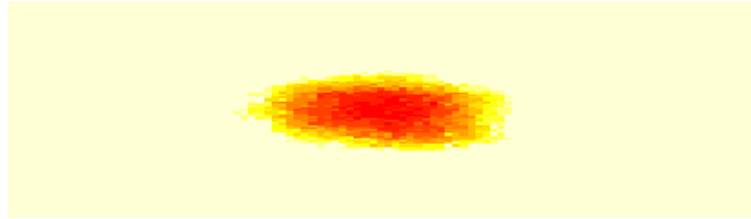
Population
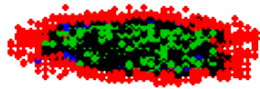
EX_glc(e)
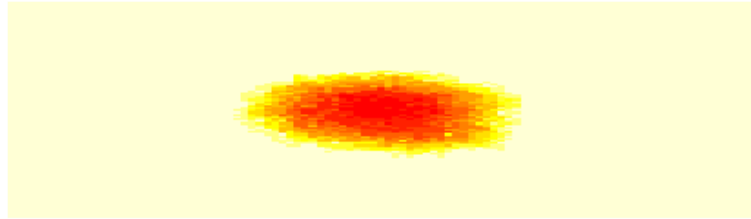
Population
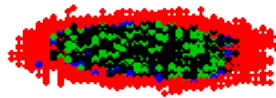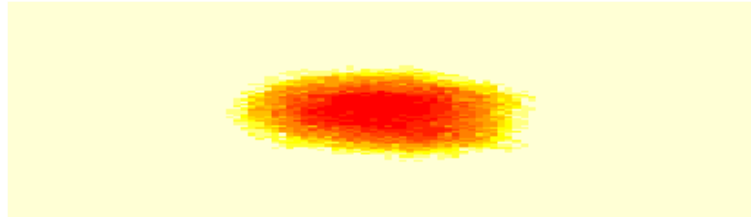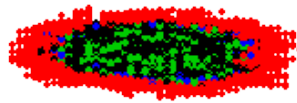
EX_glc(e)

**Population**

EX_glc(e)

Population

Now we can see that the periphery of the population has a different color than the individuals in the center. This indicates that individuals on the outside of the population use a different metabolism (respiration of glucose) than the center (fermentation of glucose and acetate). To visualize the differences of the apparent phenotypes we can use

```
minePheno(eval)
```
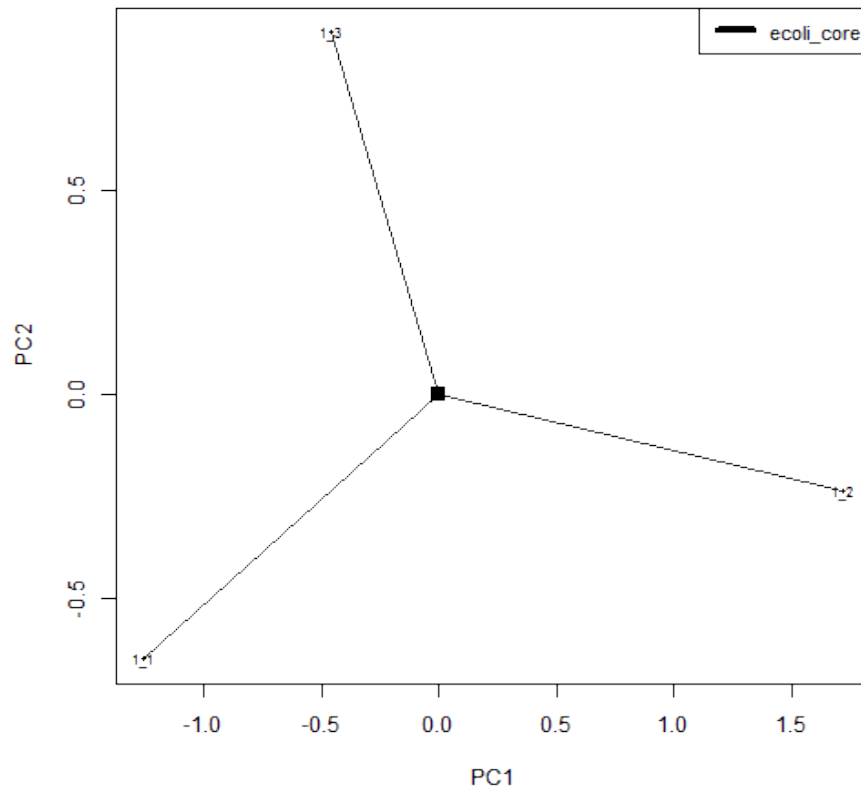
This will create a PCA plot with the similarity of the different phenotypes. If we are interested in the definition of the phenotypes we can retreive the original phenotype matrix with

```
pmat <- getPhenoMat(eval)
```

The object pmat carries now the different phenotypes which are defined by used exchange reactions within individuals on the population. A value of 1 means secretion, 2 means uptake and 0 means no usage of the substance of interest.