Multiscale Computer Setup

# Introduction

This document provides the setup information for the simulation environment. Currently the setup has to be performed manually. Future plans are to migrate to a full setup via docker based on a clean Ubuntu installation.

<https://www.docker.com/>

Until then the nodes have to be setup manually.

* all python dependencies should be handled within virtual environments
* all R clearly with the R packaging system

## Computers

The simulations are performed on the following computer hardware. In total around 40 cores are currently available for simulations

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **name** | **ip** | **cores** | **user** | **ssh access**  [ssh uname@ip] | **postgres**  **access** | **http /apache2** | **hardware settings** | **django db access** |
| home | - |  | mkoenig | - |  |  | running |  |
| sysbio2 | 10.39.32.111 | 8  (Xeon W3550 @ 3.07GHz) | mkoenig | yes | yes  postgres running | yes | running | running |
| mint | 10.39.32.106 | 8 | mkoenig | yes | - | no apache running | running | running |
| sysbio1 | 10.39.32.189 | 12  (Intel Core i7 X980 @3.33GHz) | mkoenig | yes | yes  postgres running  [check vpn] | yes | running | running |
| core | 10.39.34.27 | 12  (Intel Core i7 X980 @3.33GHz) | mkoenig | yes | - | no apache running | running | running |
| olli | 10.43.6.114 |  | matthias | yes |  | running | ? |  |
| stan | 10.43.6.115 |  | matthias | no  password missing |  | running | ? |  |

# Multiscale project

## Environment variables

Set the necessary environment variables

sudo gedit /etc/bash.bashrc

export MULTISCALE\_DB=multiscale-galactose

export MULTISCALE\_GALACTOSE=$HOME/git/multiscale-galactose

export MULTISCALE\_GALACTOSE\_RESULTS=$HOME/multiscale-galactose-results

export PYTHONPATH=${PYTHONPATH}:${MULTISCALE\_GALACTOSE}/python/multiscalepy

export PYTHONPATH=${PYTHONPATH}:${MULTISCALE\_GALACTOSE}/python/multiscalepy/multiscale/multiscalesite

export DJANGO\_SETTINGS\_MODULE=multiscalesite.settings

source /etc/bash.bashrc

## Clone project

All project code is managed on github. clone the repository in the defined locations.

cd

git clone <https://github.com/matthiaskoenig/multiscale-galactose.git> $MULTISCALE\_GALACTOSE

ln -s $MULTISCALE\_GALACTOSE ~/multiscale-galactose

mkdir $MULTISCALE\_GALACTOSE\_RESULTS

# Simulation environment

Install libraries and tools for modelling

<https://docs.google.com/document/d/1h6PU5pfogfRfevbGAYgaO8CecHD4Xg7ZKfL928Dxze0/edit#>

* libSBML
* roadRunner
* antimony

# Database: Postgres

## Server setup

The postgres server has to be running on the server.

sudo apt-get install postgresql postgresql-contrib

### User & database setup

The database needs the respective user with password set

sudo su postgres

createuser mkoenig

psql

ALTER USER mkoenig WITH SUPERUSER;

\password mkoenig

\q

Change password

sudo -u postgres psql

\password postgres

\password postgres

\q

Create database

createdb $MULTISCALE\_DB

### Allow access to database

necessary to setup the access from other computers in the postgres settings

sudo gedit /etc/postgresql/9.3/main/pg\_hba.conf

host all mkoenig samenet trust

Allow communication over TCP/IP

sudo gedit /etc/postgresql/9.3/main/postgresql.conf

listen\_addresses = '\*'

restart the database server

sudo service postgresql restart

Test the connection via

psql -h 10.39.34.27 -U $USER $MULTISCALE\_DB

## Client setup

For the interaction with the database the postgres client libraries have to be installed

sudo apt-get install postgresql-client-common postgresql-client python-dev libpq-dev pgadmin3 cmake cmake-gui

Access to the database can be tested via

psql -h dbhost -U user dbname

psql -h 10.39.34.27 -U $USER $MULTISCALE\_DB

# Django

Necessary to run at least python 2.7, so check and update if necessary.

python --version

## Django installation

sudo apt-get install llvm python-dev libpq-dev

sudo -E pip install rpy2

sudo -E pip install psycopg2==2.6.1

sudo -E pip install django==1.8.7

sudo -E pip install django-extensions django-haystack --upgrade

sudo -E pip install django-debug-toolbar --upgrade

sudo -E pip install django-enumfield --upgrade

Test installation

import django  
print(django.VERSION)

Python requirements are listed in

$MULTISCALE\_GALACTOSE/python/multiscale/requirements.txt

and can be installed via

sudo -E pip install pandas matplotlib requests sh enum34 --upgrade

**MPI**

sudo apt-get install libcr-dev mpich2 mpich2-doc

sudo -E pip install mpi4py

**h5py**

sudo apt-get install libhdf5-dev

sudo -E pip install Cython --upgrade

sudo -E pip install h5py --upgrade

## Database schema

Set the database schema in the settings.py

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.postgresql\_psycopg2',

'NAME': 'multiscale-galactose',

'USER': 'mkoenig',

'PASSWORD': '\*\*\*\*\*\*\*\*',

'HOST': 'localhost',

'PORT': '5432',

}

}

To create all tables, drop the tables via pgadmin3

cd $MULTISCALE\_GALACTOSE/python/multiscale/multiscale/multiscalesite

python manage.py syncdb

## Management tables

**Delete all tables**

manage.py sqlclear

will print the sql statement to drop all tables

Better way to manage the tables is via pgadmin3 (drop cascade)

pgadmin3

**Delete all data in all tables**

manage.py flush

returns the database to the state it was in immediately after syncdb was executed

**Create all tables as defined in the model**

manage.py syncdb

Creates the database tables for all apps in INSTALLED\_APPS whose tables have not already been created. See this page for a reference of all commands:<https://docs.djangoproject.com/en/dev/ref/django-admin/>

But look into using [south](http://south.aeracode.org/docs/about.html), it's the best way to manage your database.

## Run django database

### Console

cd $MULTISCALE\_GALACTOSE/python/multiscale/multiscale/multiscalesite

python manage.py runserver 8001

### PyCharm integration

TODO