

# H<sub>2</sub>FC

*Integrating European Infrastructure to support science and development of Hydrogen- and Fuel Cell Technologies towards European Strategy for Sustainable, Competitive and Secure Energy*

## Welcome to our H<sub>2</sub>FC Sage Tutorial!



### Document History

Issue Date	Version	Changes Made/Comments
05.03.2015	0.1	Patrick Gunzenhauser (Patrick.Gunzenhauser@kit.edu)

## Inhalt

1	Introduction .....	2
1.1	What is Sage .....	3
2	A guided Tour .....	3
2.1	First Steps in Sage .....	3
2.1.1	How to register for Sage Framework? .....	3
2.1.2	How to log-in the Sage Frame Work? .....	3
2.1.3	The user interface .....	4
2.1.4	How to publish a mathematical (research) model in Sage? .....	5
2.2	The most important Sage Commands .....	5
2.2.1	Basic arithmetic operations .....	5
2.2.2	Basic Algebra and Calculus .....	6
2.3	Basic instructions in Sage .....	7
2.3.1	Plotting - two and three dimensional .....	7
2.3.2	Python – Loops and if/else-Statement .....	8
2.3.3	Interactive Input data error .....	9
3	FAQs – or general questions on the Sage application .....	9

## 1 Introduction

Sage is a free open source mathematics software system (based on Python) being licensed under the GPL (General Public License). The H2FC team has implemented an own H2FC Sage framework, especially adjusted regarding hydrogen and fuel cell models. The aim of this Tutorial is to set out a general overview of our Sage Framework. For more details on sage please look at <http://sagemath.org/>, for python at <https://www.python.org/> or use some introductory books based on Python.

The tutorial aims to make you familiar with our Sage Framework. If you are interested in gaining a short overview on sage, don't hesitate to follow the steps of this tutorial which will last only a few minutes. In this tutorial special functions and commands in sage will not be mentioned and have to be looked-up in the reference manual of sage (either look at [www.sagemath.org](http://www.sagemath.org) (→Help/Documentation→Reference Manual) or on our Sage Notebook (→Help → Reference Manual).

## 1.1 What is Sage

Being composed on existing open-source packages such as [NumPy](#), [matplotlib](#), [Maxima](#) and many more, the software can be used for number theory, algebra and geometry computations. In general it can be said that sage supports research and teaching in algebra, geometry, number theory, cryptography, numerical computation and related areas.

## 2 A guided Tour

This part of the tutorial wants to set out a guided route from registration to the first steps of creating your own models within our Sage Framework.

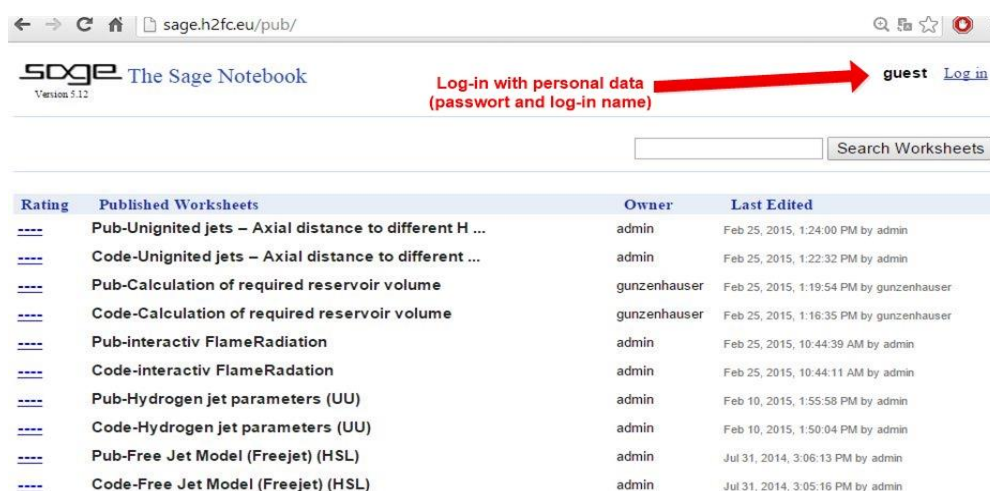
### 2.1 First Steps in Sage

#### 2.1.1 How to register for Sage Framework?

In order to get access to our Sage Server, please contact [sharepoint@h2fc.eu](mailto:sharepoint@h2fc.eu) or register on our webpage <http://h2fc.eu/sageserver>. Please be aware that access will be granted within 48 hours.

#### 2.1.2 How to log-in the Sage Frame Work?

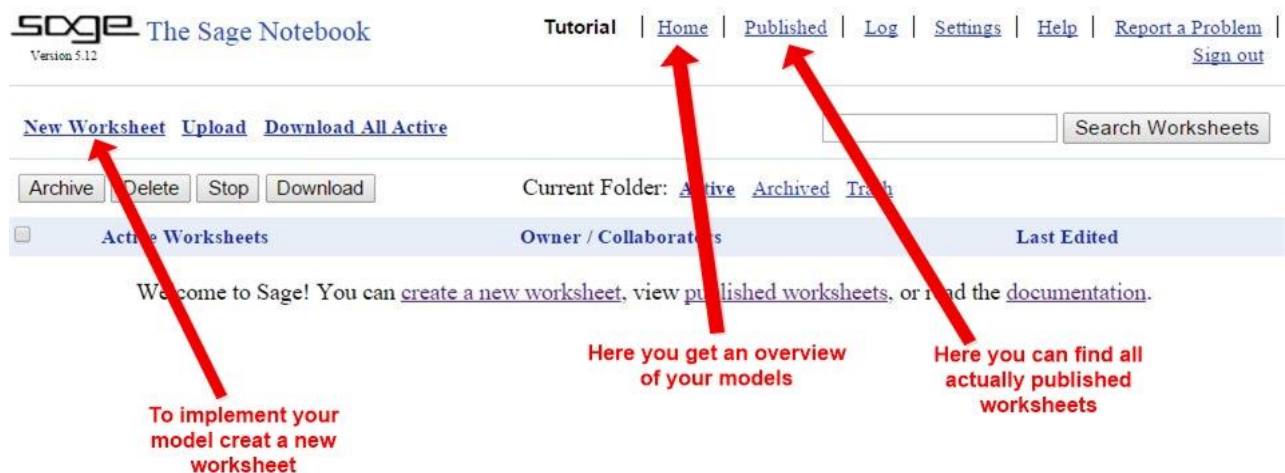
After registration you will perceive your personal log-in data consisting of a password and your log-in name (in general family name). Then you can log-in ether on the published worksheet page (<http://sage.h2fc.eu/pub/>) or on the start page of Sage notebook (<http://sage.h2fc.eu/>).



Rating	Published Worksheets	Owner	Last Edited
----	Pub-Unignited jets – Axial distance to different H ...	admin	Feb 25, 2015, 1:24:00 PM by admin
----	Code-Unignited jets – Axial distance to different ...	admin	Feb 25, 2015, 1:22:32 PM by admin
----	Pub-Calculation of required reservoir volume	gunzenhauser	Feb 25, 2015, 1:19:54 PM by gunzenhauser
----	Code-Calculation of required reservoir volume	gunzenhauser	Feb 25, 2015, 1:16:35 PM by gunzenhauser
----	Pub-interactive FlameRadiation	admin	Feb 25, 2015, 10:44:39 AM by admin
----	Code-interactive FlameRadiation	admin	Feb 25, 2015, 10:44:11 AM by admin
----	Pub-Hydrogen jet parameters (UU)	admin	Feb 10, 2015, 1:55:58 PM by admin
----	Code-Hydrogen jet parameters (UU)	admin	Feb 10, 2015, 1:50:04 PM by admin
----	Pub-Free Jet Model (Freejet) (HSL)	admin	Jul 31, 2014, 3:06:13 PM by admin
----	Code-Free Jet Model (Freejet) (HSL)	admin	Jul 31, 2014, 3:05:16 PM by admin

### 2.1.3 The user interface

The user interface of sage is called Sage Notebook. The Sage Notebook provides different buttons in order to gain access to general sage function. The button “home” is leading you to an overview window of your own developed models. The button “Published” navigates you to the surface of all published worksheets. The Sage Notebook is a collection of worksheets. You can create your own worksheet by using the button “new worksheet” in the left-hand panel.



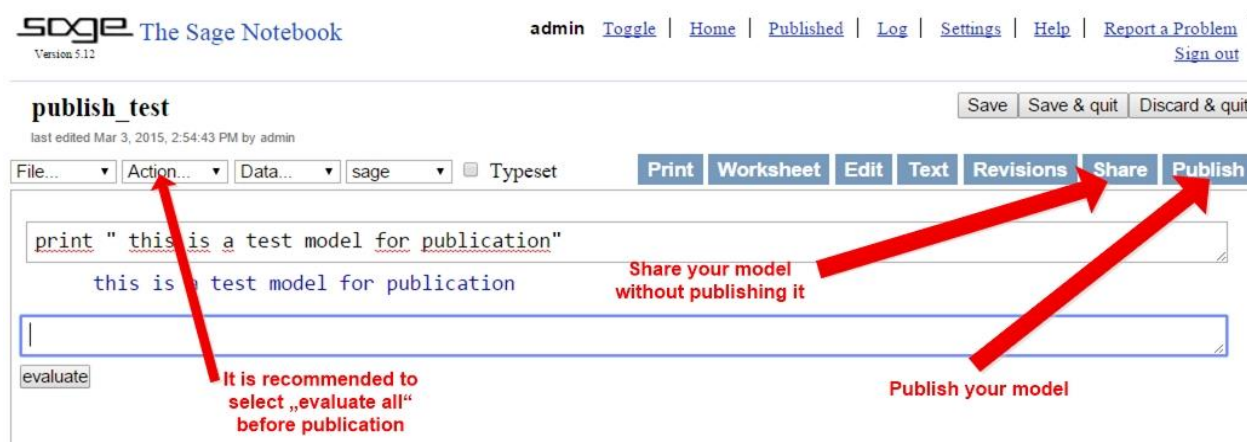
A Sage Worksheet consists of several input boxes in which you can type just one or various instructions. This enables you to use sage as a usual pocket calculator. After typing some instructions in the input boxes the commands have to be evaluated. The result appears below the text box, while a new one turns up. As you can see just the last result will be shown after evaluation. For displaying former results use the “Print” command.



Above the text boxes you will find numerous buttons, such as save, copy, load etc. One of the most important one is the “Save&Quit” button. This one saves your worksheet and leads you back to the overview page.

### 2.1.4 How to publish a mathematical (research) model in Sage?

In order to publish your models of hydrogen and fuel cell research you have to implement them first in a worksheet within the Sage notebook. Before you publish your model it is recommended to click the action field and select “evaluate all” to evaluate your code.



**publish\_test**  
last edited Mar 3, 2015, 2:54:43 PM by admin

File... Action... Data... sage Typeset

Print Worksheet Edit Text Revisions Share Publish

print " this is a test model for publication"  
this is a test model for publication

evaluate

It is recommended to select „evaluate all“ before publication

Share your model without publishing it

Publish your model

## 2.2 The most important Sage Commands

### 2.2.1 Basic arithmetic operations

As already mentioned Sage can be used as an usual pocket calculator. No matter on how complex mathematical operations can be, you can also just use the basic arithmetic operations in Sage (summation, subtraction, multiplication, division, and exponentiation).

```
a=2
b=3

print a+b
print a-b
print a*b
print a/b
print a^b
```

```
5
-1
6
2/3
8
```

### 2.2.2 Basic Algebra and Calculus

Sage can perform various computations related to basic algebra and calculus. For example: finding solutions to equations, differentiation, and integration.

In all these examples, it is important to note that the variables in the functions are defined to be `var(...)`.

For example: `sage: u = var('u')`

`sage: diff(sin(u), u)`

#### ➤ Solving Equations

The solve function solves equations. If you want to use this function, you should firstly specify some variables and then the arguments in order to solve the equation.

`sage: x = var('x')`

`sage: solve(x^2 + 3*x + 2, x)`

`x = -2, x = -1`

#### ➤ Solving Differential Equations

You can use Sage to solve ordinary differential equations. For example the equation:  $x' + x - 1 = 0$

`sage: t = var('t') # define a variable t`

`sage: x = function('x',t) # define x to be a function of that variable`

`sage: DE = diff(x, t) + x - 1`

`sage: desolve(DE, [x,t])`

#### ➤ Differentiation and Integration

Sage knows how to differentiate and integrate many functions. For example, to differentiate  $\sin(x)$  with respect to  $x$ , do the following:

`sage: x = var('x')`

`sage: diff(sin(x), x) cos(x)`

For example to integrate the following function:  $\int x \sin(x^2) dx$

`sage: integral(x*sin(x^2), x)`

`-1/2*cos(x^2)`

## 2.3 Basic instructions in Sage

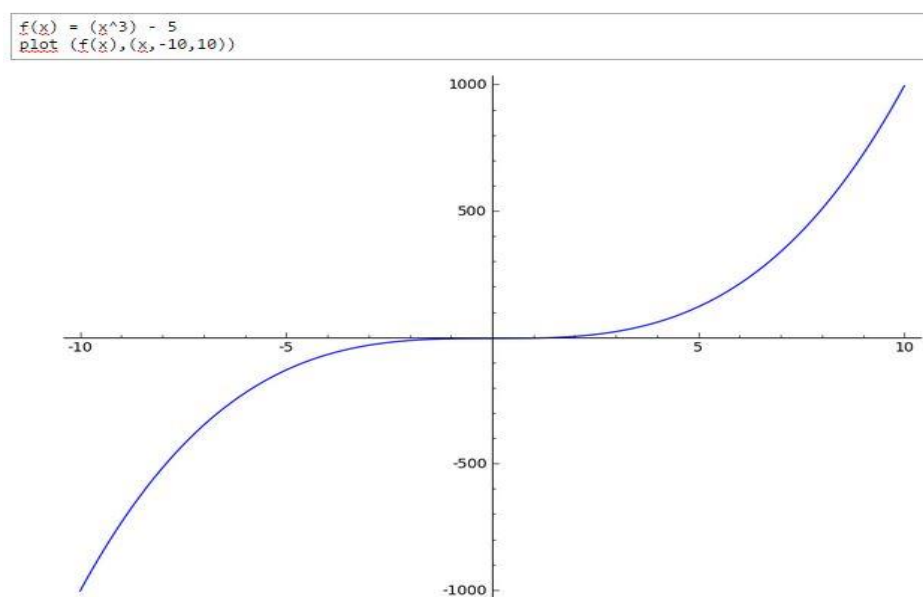
In this section of the tutorial you will be given some basic hints, for example how to use python language in Sage.

### 2.3.1 Plotting - two and three dimensional

In Sage you can produce two or three-dimensional plots (see graphics below).

In **two dimensions**, Sage can draw circles, lines, and polygons, plots of functions in coordinates and also polar plots, contour plots and vector field plots. We present examples of some of these here.

If you want to draw the graph of on function, it is recommended to use the plot command. By choosing  $f(x) = x^3 - 5$  and executing plot( $f(x), (x, -10, 10)$ ) sage plots the following graph:



*Another example:* This command produces a yellow circle of radius 1, centered at the origin:

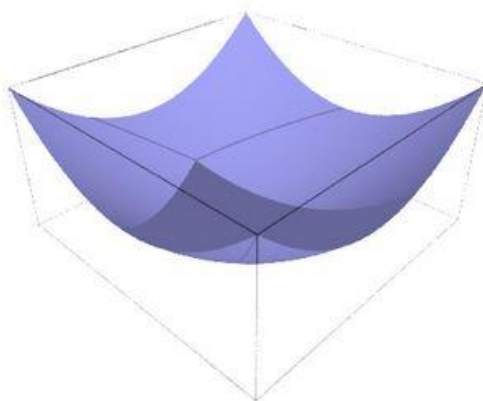
**sage:** circle((0,0), 1, rgbcolor=(1,1,0))

Sage can also be used to create **three-dimensional** plots. In this case you have to work with the “plot3d” command.

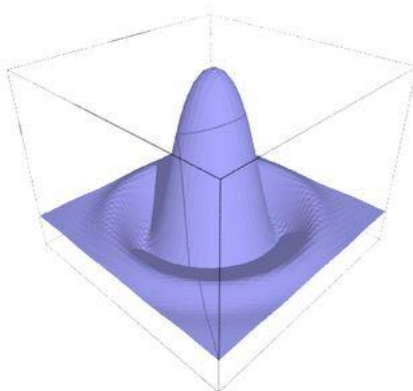
If you want to plot a 3d-function you have to start with defining the variables. Then you can define your function and the section of the graph within the plot3d command:

*For example:*

1.) **sage:** `var('x y'); plot3d(x^2 + y^2, (x,-2,2), (y,-2,2))`



2.) **sage:** `var('x y'); plot3d(cos((x^2+y^2)*2) * e^-(x^2+y^2), (x,-2,2), (y,-2,2)).show(aspect_ratio=(1,1,2.5))`



### 2.3.2 Python – Loops and if/else-Statement

Due to being based on python, sage is a programming language as well. This part of the tutorial will give you only some hints. For more details you can also use tutorials of Python or Sage as well. If you use a “for-loop” or “if/else” statements please consider that indentions of your programming code after the loops determine your program.



There aren't curly brackets to open up a loop as it is usual in other programming languages. See one example below:

```
for i in range (10):  
    if i > 2  
        print "i ist größer 2"
```

### 2.3.3 Interactive Input data error

If you want to have a user interface of your mathematical models, you have the chance to use different interactive input data errors. In order to implement such fields in sage you can use the "@interact" command. Sage provides different possibilities of creating input boxes. One example is shown below. It is an interactive box with a slider to change intake parameters.

```
sage: interact  
def _(a=5, y=(0..20)): print a + y
```



## 3 FAQs – or general questions on the Sage application

- Can I have access to published Sage models without registration?
  - Yes, you can take a look at the currently published models in our sage framework without being registered. For this, please take a look at: <http://sage.h2fc.eu/pub/>
- Can I use other components such as mathematica, matlab or maple?
  - Yes, in general this would be possible. Sage provides different toolkits for mathematical application, if they are installed. Actually there are only the free toolkits installed. If you have any specific questions regarding

mathematical tools which you want to use on our sage Framework, please contact the administration.

- What should I do if my personal log-in does not work?
  - You should please first note the capitalization and remove blanks at the end of your log-in name and password. If the log-in is still impossible, please contact [sharepoint@h2fc.eu](mailto:sharepoint@h2fc.eu) again and describe your problem in short, to get personal support.
- Can I share my models only with a limited number of people without publishing them?
  - Yes, you can share your worksheets without publication. For this, just click the button “share” instead of “publish” in you worksheet and select the people you want to transmit access to your worksheets.
- Can I use different Test fields within one worksheet in order to spread my code?
  - Yes it is possible and gives you the advantage of evaluating only some sections of your developed code.
- How can I get help for Sage commands?
  - Sage contains a comprehensive built-in documentation, accessible by typing the name of the search function followed by a question mark in the evaluation box.  
e.g. **sage: tan?**

***If you have further questions according to our Sage Framework please don't hesitate to contact the administration! For more information we recommend:***  
***<http://www.sagemath.org/pdf/en/tutorial/SageTutorial.pdf>***  
***or: <http://www.sagemath.org/doc/tutorial/>***