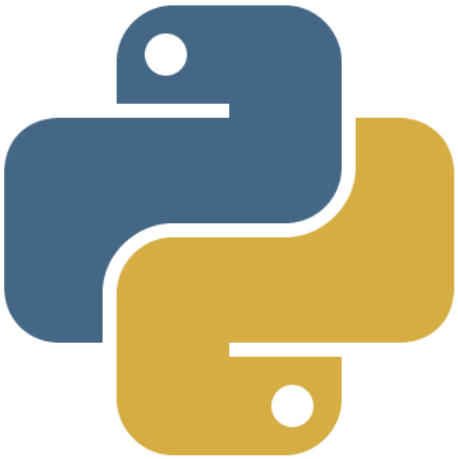
PYTHON



Python é uma linguagem de programação de *alto nível*, *interpretada*, de *script*, *imperativa*, *orientada a objetos*, *funcional*, de *tipagem dinâmica e forte*.
A linguagem foi projetada com a filosofia de enfatizar a importância do esforço do programador sobre o esforço computacional. Prioriza a legibilidade do código sobre a velocidade ou expressividade. Combina uma sintaxe concisa e clara com os recursos poderosos de sua biblioteca padrão.

Abaixo iremos pormenorizar cada um de seus atributos:

- **Linguagem Interpretada** – é uma linguagem de programação em que o código fonte nessa linguagem é executado por um programa de computador chamado interpretador e em seguida é executado pelo sistema operacional ou processador.
  Ou seja, de uma forma ela é semelhante à linguagem HTML que é interpretada no navegador do usuário no ato do acesso.
- **Linguagem de Script** – é uma linguagem que é executada no interior de programas ou de outras linguagem de programação, não se restringindo a estes ambientes.
  Ou seja, podem estender as funcionalidades de um programa sem contudo alterá-lo e/ou controla-lo.
- **Linguagem Imperativa** – que descreve o programa como um conjunto de ações, enunciados ou comandos que mudam o estado de um programa dizendo a cada passo o que deve ser executado ou interpretado.
  Ou seja, está relacionado ao tempo verbal imperativo, onde o programador diz ao computador "faça isso, depois isso, depois aquilo".
- **Orientada a Objetos** – é um modelo de análise, projeto e programação baseado na composição e interação entre diversas unidades chamadas de objetos.
  Os objetos são partes de código independentes e funcionais que podem ser aproveitadas conforme o desejo e necessidade do programador. Depois de definidos, implementa-se um conjunto de classes que definem os objetos presentes no sistema e cada classe determina o comportamento e estados possíveis de seus objetos assim como o relacionamento com outros objetos.

- **Programação Funcional** – é uma abordagem de programação que enfatiza a aplicação de funções. A ideia básica é utilizar-se de funções predefinidas que evitam estados ou dados mutáveis. No caso de Python, as inúmeras funções preexistentes auxiliam desde a simples formatação de texto até a resolução de cálculos, concatenação de valores e até operações com arquivos.
  Este tipo de abordagem contrasta em parte com a programação imperativa que, além das funções, já possui construções mais complexas. Desta forma, Python não é considerada estritamente funcional, pois além de funções utilizadas no ato da interpretação, também se utiliza dessas construções imperativas.
- **Tipo forte** – Linguagem implementadas com tipificação forte exigem que o tipo de dado de um valor seja do mesmo tipo da variável ao qual este valor será atribuído. Ou seja, se declararmos uma variável como inteira, somente será aceita a entrada de valores inteiros para esta variável.
- **Tipo Dinâmico** – A verificação de um dado é feita de forma dinâmica, ou seja, em tempo de execução – característica de uma linguagem interpretada.

.

Construções

Construções em Python incluem:

- Estrutura de seleção (*If, else elif*)
- Estrutura de repetição (*for, while*)
- Construção de classes (*class*)
- Construção de sub-rotinas (*def*)
- Construção de escopo (*with*)

.

Tipos de dado

A tipagem de Python é forte, pois os valores e objetos têm tipos bem definidos e não sofrem coerções como em outras linguagens. São disponibilizados diversos tipos de dados nativos:

- *str, unicode* – uma cadeia de caracteres imutável
- *list* – lista heterogênea mutável
- *tuple* – tupla imutável (lista ordenada de n elementos)
- *set, frozenset* – conjunto não ordenado, não contem elementos duplicados
- *dic*t – conjunto associativo

- *int* – número de precisão fixa
- *float* – ponto flutuante
- *complex* – número complexo
- *bool* – booleano (1 ou 0, verdadeiro ou falso)

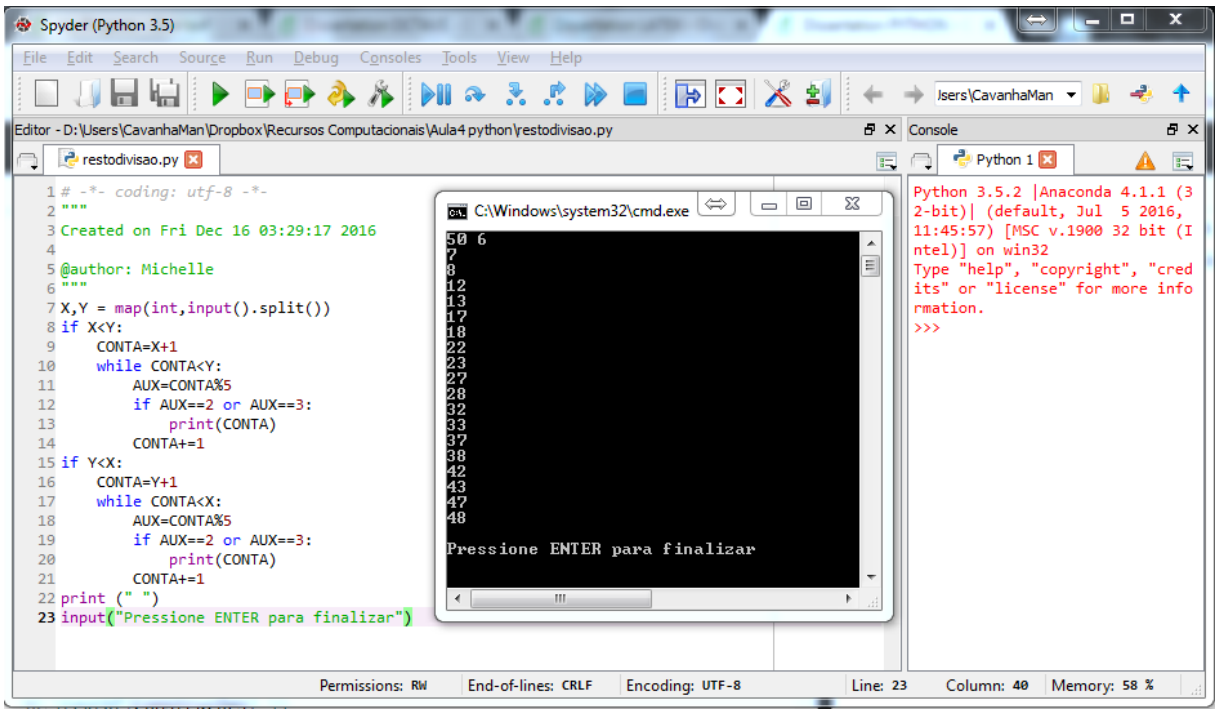
Python também permite a definição dos tipos de dados próprios através de classes.

Palavras Reservadas

O Python 2.5.2 define as seguintes 31 palavras reservadas, que não podem ser utilizadas como um identificador por fazerem parte da gramática da linguagem:

in	*def*	*assert*	*raise*
is	*not*	*global*	*continue*
if	*del*	*break*	*lambda*
as	*and*	*from*	*return*
or	*else*	*exec*	*finally*
try	*pass*	*yield*	*except*
elif	*with*	*class*	*import*
for	*while*	*print*	

Tela do compilador Python Spyder do Anaconda:



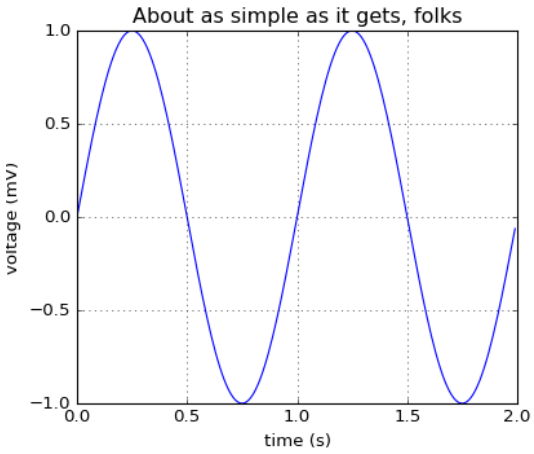
MATPLOTLIB

Matplotlib é uma biblioteca de plotagem 2D, ou seja é uma coleção de sub-rotinas e funções utilizadas em **Python** no desenvolvimento de gráficos em duas dimensões. Essa biblioteca consegue gerar figuras de qualidade em uma grande variedade de formatos e de ambientes interativos em diversas plataformas.

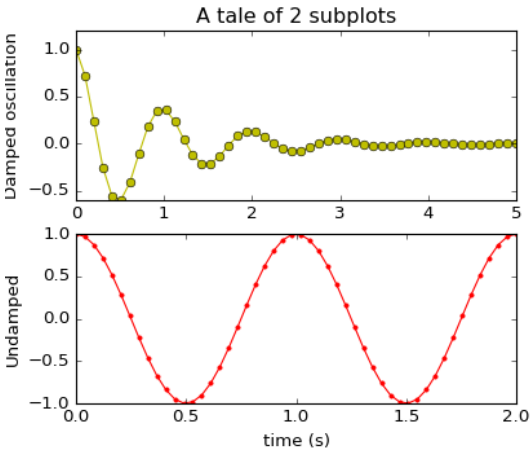
O objetivo da biblioteca é tornar fácil as tarefas rotineiras e tornar possíveis as tarefas mais difíceis. Ela permite gerar inúmeros modelos computacionais com apenas algumas linhas de código. Sua interface é simples e semelhante ao Matlab.

Segue a seguir alguns exemplos de plotagem geradas em **matplotlib**.

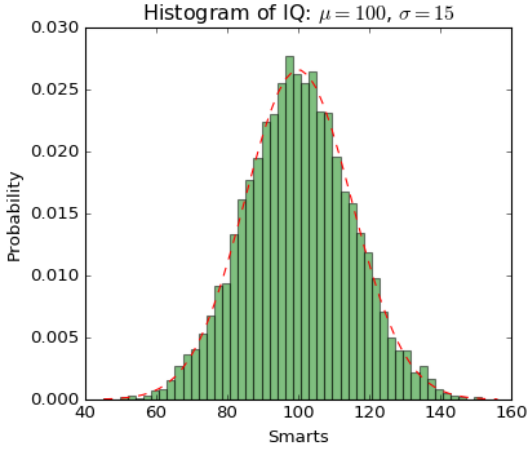
Plotagem simples:



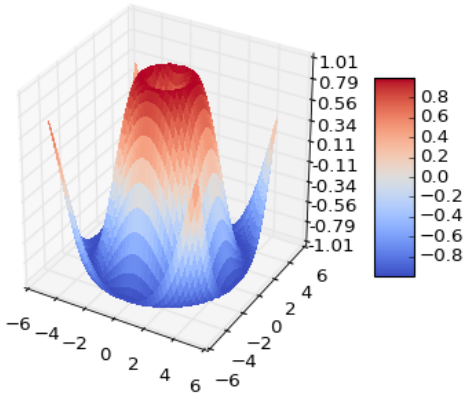
Plotagem com múltiplos eixos gerado pelo comando subplot():



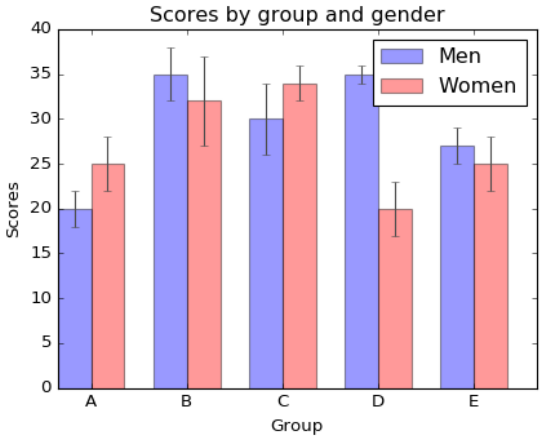
Plotagem de histograma (distribuição de frequência) gerada com o comando *hist()*:



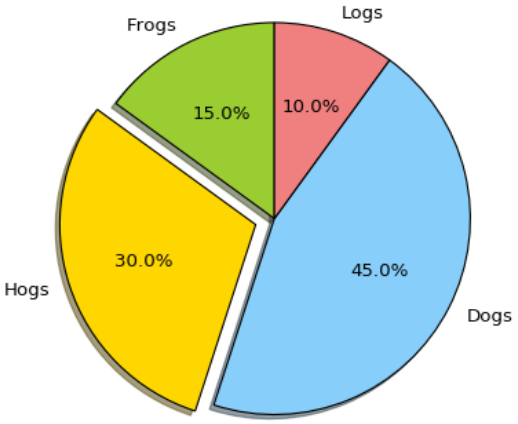
Gráficos 3D simples utilizando a ferramenta *mplot3d*:



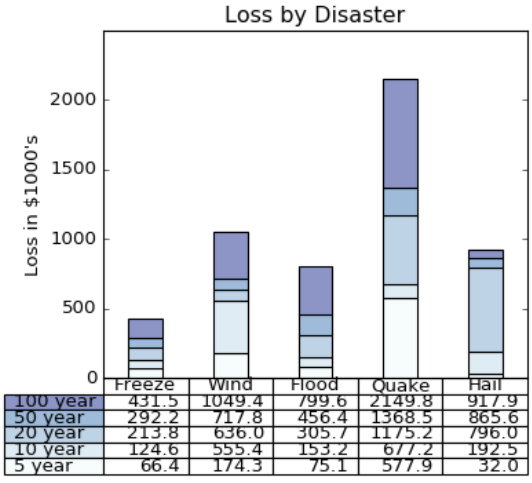
Gráficos em barras – que são facilmente gerados com o comando *bar()*:



Gráficos em pizza (ou torta) – gerados facilmente com o comando *pie()*:

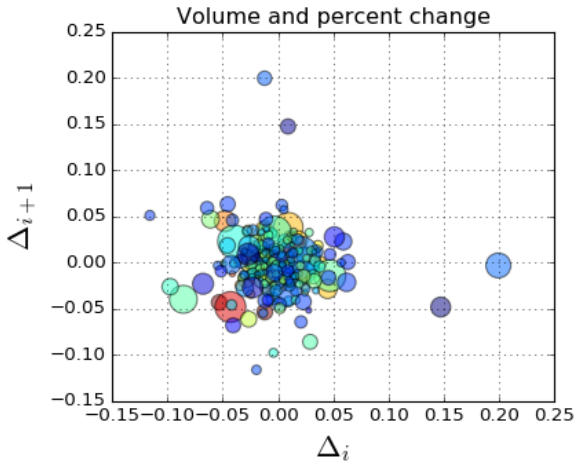


Gráficos com tabelas – gerados com o comando *table()*:

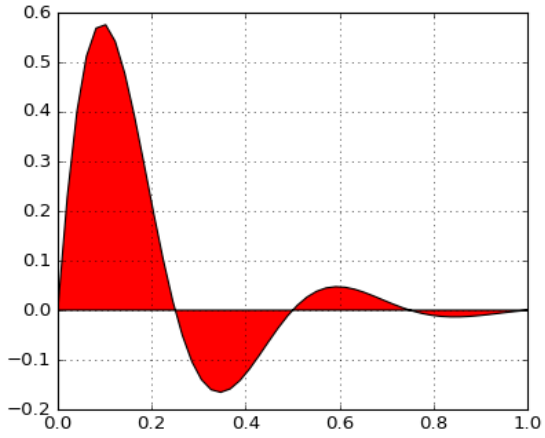


	Freeze	Wind	Flood	Quake	Hail
100 year	431.5	1049.4	799.6	2149.8	917.9
50 year	292.2	717.8	456.4	1368.5	865.6
20 year	213.8	636.0	305.7	1175.2	796.0
10 year	124.6	555.4	153.2	677.2	192.5
5 year	66.4	174.3	75.1	577.9	32.0

Gráfico de dispersão, gerado com o comando *scatter()*:

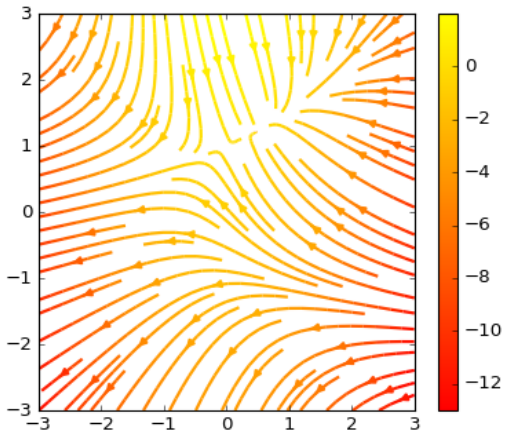


Plotagem de curvas e polígonos preenchidos com o comando *fill()*:

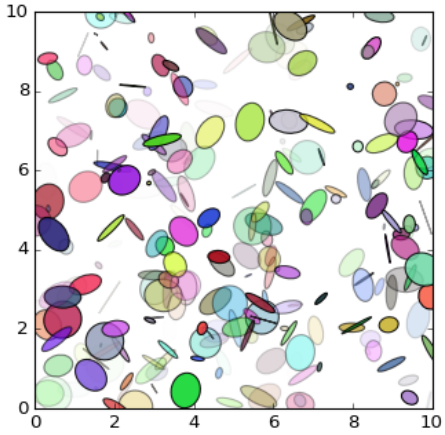


Plotagem de dados de data com intervalos maiores ou menores:

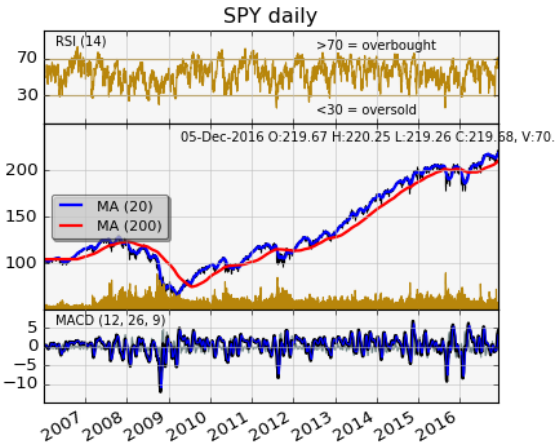
Linha de corrente gerada com o comando *streamplot()*:



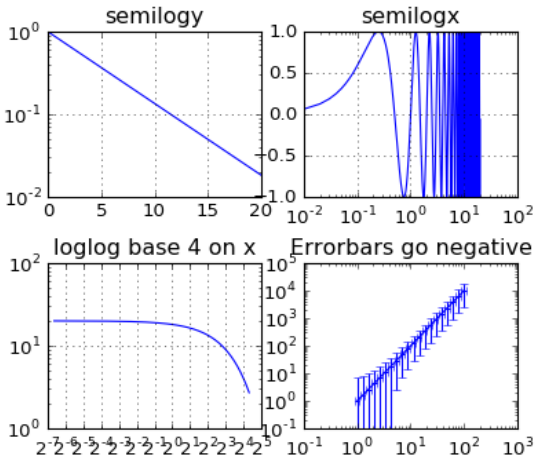
Elipses:



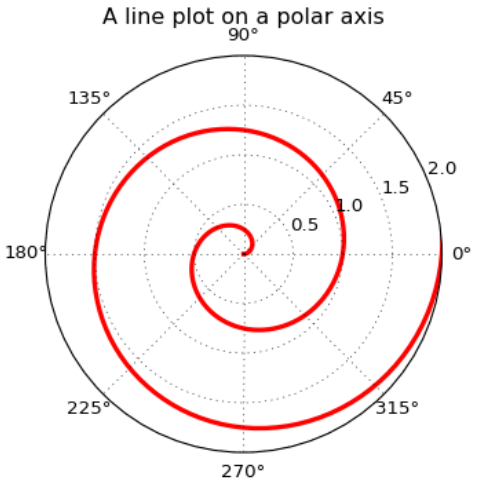
Gráficos financeiros:



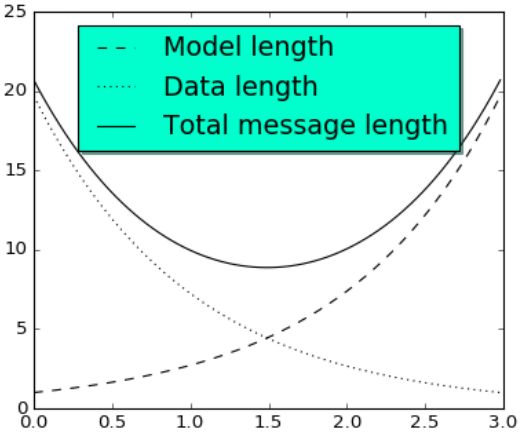
Funções simples para criação de plotagens logarítmicas – *semilogx()*, *semilogy()* e *loglog()*:



Gráficos polares com o comando *polar()*:



Comando *legend()* para gerar automaticamente legendas:



MATH

Este módulo está sempre disponível e provê acesso às funções matemáticas definidas pela *linguagem C*.
As funções são:

- Funções da Teoria dos Números e de Representação
- Funções de Potência e Logarítmicas
- Funções Trigonométricas
- Conversão Angular
- Funções Hiperbolicas
- Além de Funções Especiais e Constantes

Em Python estas funções não utilizarão números complexos. Neste caso utilizaremos as mesmas funções porém na biblioteca *cmath*.