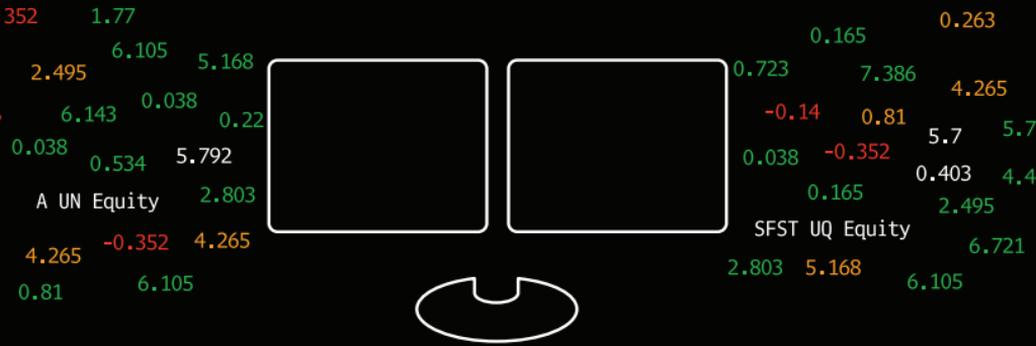


A systematic approach to company research.

Build, test and share bespoke factor strategies with BQuant.

Bloomberg

Introducing BQuant, Bloomberg's quantitative analytics platform.



Buy-side company research analysts are faced with managing increasing amounts of complex data to evaluate investment strategies and better understand relative return potential.

Bloomberg has built BQuant, an interactive development tool that enables users to build, test and share research – with faster time to market.

The BQuant environment is built on open source tools like Python and JupyterLab and is fully integrated with the Bloomberg Terminal®.

Manage multiple complex datasets in one integrated solution.

BQL, Bloomberg's new query language, is a powerful tool that enables users to access the rich universe of Bloomberg data as well as perform computations such as screening and aggregation.

BQuant uses BQL to enable users to leverage Bloomberg's analytics and infrastructure for tasks such as factor research, that can ultimately be shared as interactive visualizations with other Bloomberg users to make better investment decisions.

BQuant

Bloomberg's quantitative analytics platform.



BQL
Bloomberg data



Analytics &
infrastructure



Shareable
interactive
visualizations

Screen and aggregate data using Bloomberg's infrastructure.

BQL allows you to retrieve curated data on the Bloomberg Terminal. The data is normalized, aligned and linked across data sets.

Explore relationships between different data sets with ease including point-in-time reported and estimated company financials.

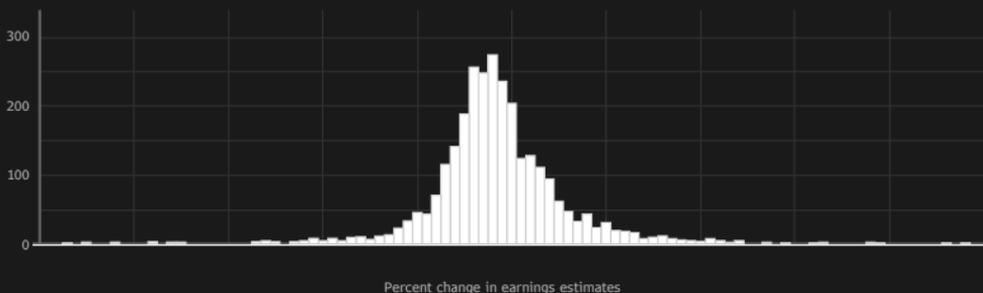
BQL Code

```
pct_chg(is_eps(dates=range(-3m, 0d), fpo=1))
```

Distribution of percent change in earnings estimates for the Russell 3000 index.

Universe: RAY Index BQL Expression: pct_chg(is_eps(dates=range(-3m, 0d), fpo=1))

of securities



Universe	▲ Count	Average	St Dev	Min	Max
Universe	2891	1.4077	3.4325	-18.28	22.74

Access pre-built models or build your own with open source tools.

Bloomberg provides sophisticated libraries for activities such as factor research and factor scoring.

Pre-built models using these libraries can be customized, or you can build your own analyses using our underlying engines and open source tools.

BQFactor

```
from bqfactor import Factor
my_factor = Factor(bq.data.is_eps(dates=range('-3m', '0d')).pct_chg())
my_factor().analyze(universe, start='2008-01-31', end='2017-12-31',
freq='m', n_quantiles=5)
```

```
import bqfactor
import bq
bq = bq.Service()

dates = bq.func.range('2018-01-01', '2018-12-31', frq='m')
universe = bq.univ.members("RAY Index")
context = bqfactor.create_analysis_context(universe, dates, bq)
eps_momentum_factor = bq.data.is_eps(dates=bq.func.range('-12m', '0d'), fill='prev').pct_chg()
```

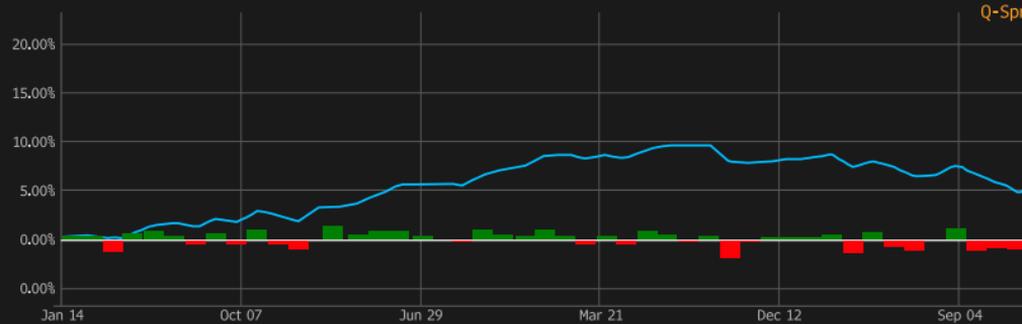
Import the BQL library to access data. Import the pandas and numpy libraries.

Visualize, share and publish interactive models.

Leverage our charting libraries to visualize data and understand your research results. With BQuant, you can share your models to collaborate with quant colleagues and also publish interactive apps to other Bloomberg users at your firm as Launchpad apps. The published interactive models are delivered as an app via Launchpad.

Analyzing Q-Spread

```
bqv.QSpreadPlot(qspread_periodic, title="Q-Spread Return", tick_format='.2%', num_ticks=12, colors=['DeepSkyBlue'])
```



The screenshot shows the BQuant Platform interface. The top menu includes Project, File, Edit, View, Insert, Cell, Kernel, Wolgata, and Help. The main workspace is divided into several panels:

- Project Library:** Displays the current project "bquant factor research" with options for Project, File, Edit, View, Run, Kernel, tabs, Settings, and Help.
- SEARCH:** A search bar for finding files and kernels.
- BLOOMBERG:** A sidebar with options for BQL Editor, BQL Help, and CONSOLE.
- CONSOLE:** A panel for running and managing kernels, including options like Change Kernel, Clear Console Cells, Close and Shutdown, Insert Line Break, Interrupt Kernel, Restart Kernel, Run Cell (forced), Run Cell (unforced), and Show All Kernel Activity.
- FILE OPERATIONS:** A panel for file management, including Autosave Documents, Close All, Close Getting St..., Close Other Tabs, Close Tabs to Right, Open From Path..., Reload from Disk, Revert to Checkpoint, Save, Save As..., and Index As...
- HELP:** A panel with links to BQL Basics, BQL Search, bqplot Reference, BQuant Help, BQuant Packages, and bqviz Reference.

The main workspace contains a code editor with the following Python code:

```
# Init display
# Get data for the default ticker

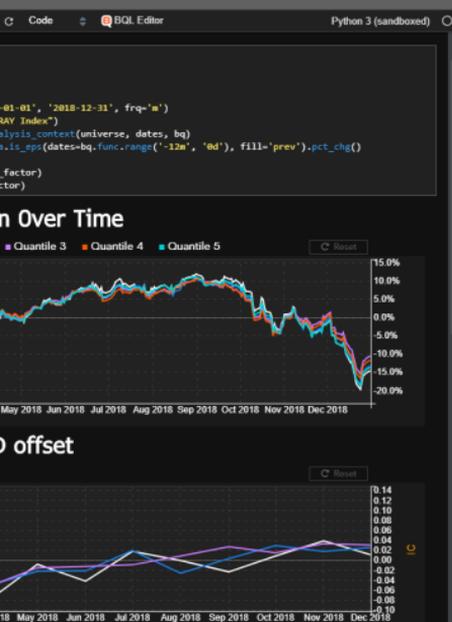
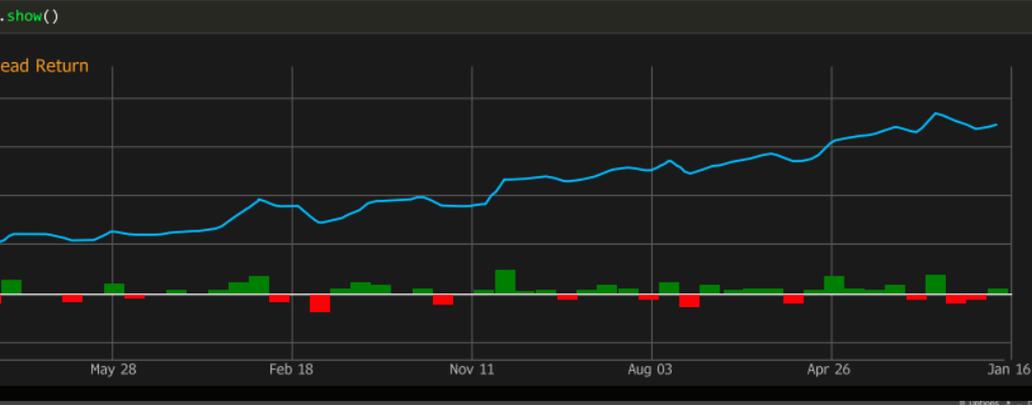
form = VBox()
HBox([ctrl_ticker_ac, ctrl_button],
HBox([], layout=Layout(margin=20px)),
HBox([ctrl_txt_ValueParam, ctrl_txt_Grwt, ctrl_txt_Mom])
)

display(form)
 sbar = StatsBar()
 dg=Grid()
 Calc_Scoring()
 ctrl_button.on_click(Calc_Scoring)
```

Below the code editor, there is a "Run" button and a "RAY Index" label. The bottom part of the interface shows a bar chart with categories: Real Estate, Materials, Information Technology, and Health Care. The chart shows varying values for each category, with Information Technology having the highest value.

BQuant allows you to program your Bloomberg in a systematic way – without the need to invest in a new environment of your own.

Take your investment analysis to the next level – from normalizing and analyzing the data to visualizing and disseminating the analysis – in one integrated solution.



Getting Started With BQL

Introduction

- BQL Editor
- Current Data
- Historical Data
- Data Analysis
- Multi-ID Universes
- Universe Filtering
- Grouping by Sector, Region, Etc.
- Advanced Query Notation

Introduction

The Bloomberg Query Language (BQL) is a powerful API that lets you retrieve Bloomberg data and transform it directly in the Bloomberg Cloud. With BQL, you can apply calculations, statistical analysis, screening, and more to Bloomberg's normalized, point-in-time data before ingesting it in the BQNT environment.

A simple BQL query contains two required clauses:

- get()**: The data you want to retrieve and any transformations or groupings you want to apply to it. For example, you can write an expression that calculates the average profit margin by sector.
- for()**: A universe containing the IDs of one or more securities, countries, or other entities on which you want to base your query. For example, you can write an expression that filters an index's members for companies with earnings per share (EPS) greater than zero.

Additional clauses are optional and discussed in [Advanced Query Notation](#).

BQL Editor

The BQL Editor guides you through writing a query. Just start typing in the `get()` or `for()` field in plain English. As you type, a drop-down menu appears and offers you suggestions. You can click a suggestion to add it to your query.

If you want help on anything in the drop-down menu, just click its  icon.

When you want to run a query to preview its results, click the **Run** button. The output appears directly below the BQL Editor fields.

Current Data

The most basic query you can write returns one point of current data for a single security, country, or

Take the next step.

Bloomberg has a team of specialists to discuss your specific use case needs. To get started with BQuant, please contact your Bloomberg representative.

Beijing

+86 10 6649 7500

Dubai

+971 4 364 1000

Frankfurt

+49 69 9204 1210

Hong Kong

+852 2977 6000

London

+44 20 7330 7500

Mumbai

+91 22 6120 3600

New York

+1 212 318 2000

San Francisco

+1 415 912 2960

São Paulo

+55 11 2395 9000

Singapore

+65 6212 1000

Sydney

+61 2 9777 8600

Tokyo

+81 3 3201 8900

[bloomberg.com/professional](https://www.bloomberg.com/professional)