ONEMARKETDATA

Data Management Challenges in Quant Research:

Solutions from OneTick and R

Maria Belianina, Ph.D.
Director, Pre-Sales Engineering Support

ONEMARKETDATA

Contents

- **Data Management Challenges in Quant Research**
- **R** and **OneTick: Addressing the challenges**
  - What is OneTick
  - Leveraging strong points of OneTick and R
  - OneTick data pre-processing and **R** analytics
  - **R** ↔ **OneTick** integration: 2 methods
- **Examples**
  - Bring OneTick Queries or Raw Data into **R** environment
  - Use **R** Functions in OneTick queries

ONEMARKETDATA

Data Management Challenges

Getting, storing and processing market data for quantitative research became more demanding as we face:

- **Increasing data granularity**
  - Daily to intraday
  - Milli → Micro → Nano → Picoseconds…
- **Data cleansing** challenges
- **Complexity** of data and data consolidation
  - Order book analytics
  - Order books vs Trades vs Custom and Summary data
- Increasing data **volumes**
- **Reference data** (corporate actions, name changes, continuous contracts, etc)
- **Security master** maintenance
- Database **schema changes**

Reporting and **analytics** *vs* **all of the above**

ONEMARKETDATA

R AND ONETICK: ADDRESSING THE CHALLENGES

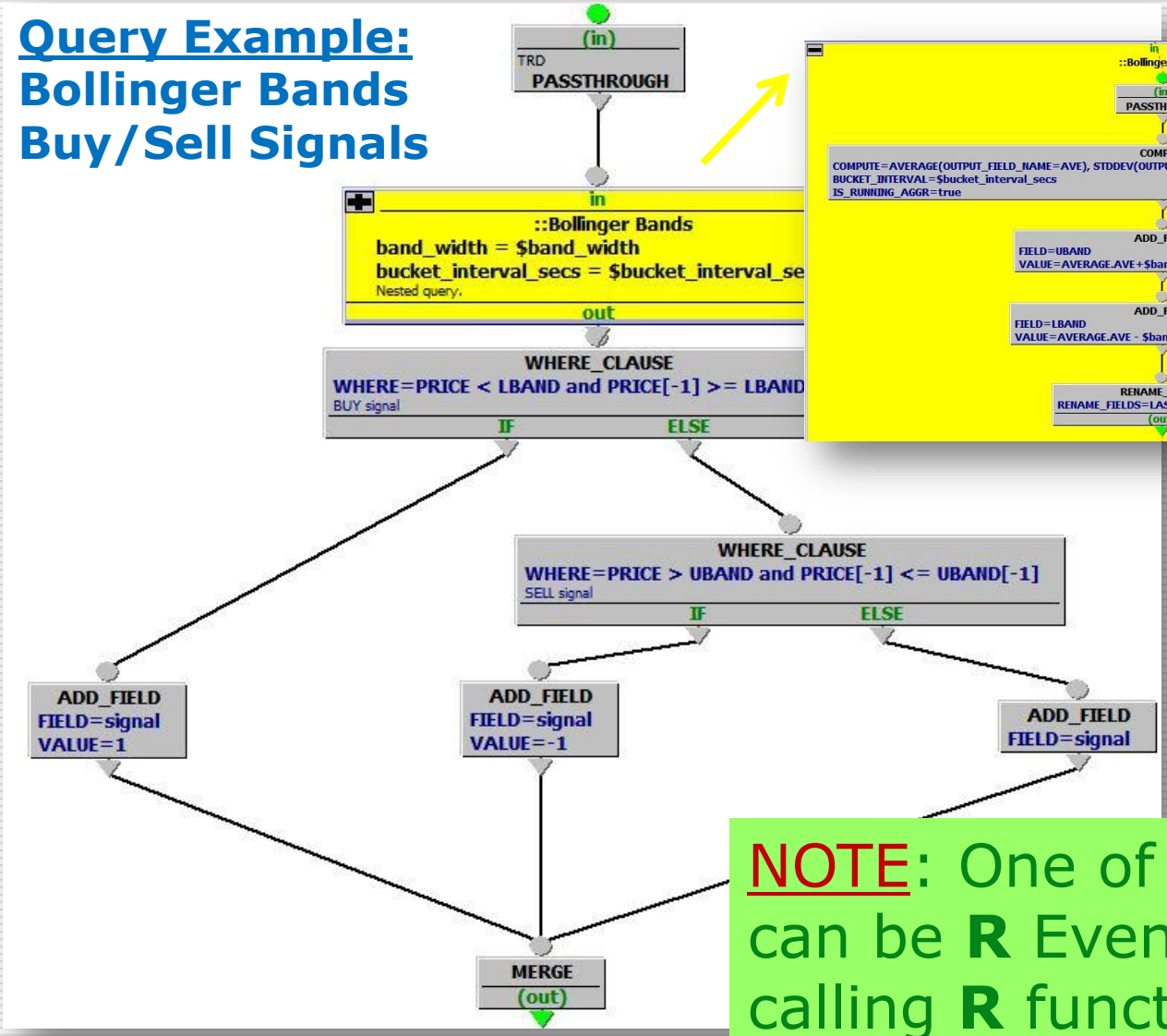
ONEMARKETDATA

What is OneTick: Overview

- ✓ **Data feed adapters**, collectors and normalization
- ✓ **Unlimited time series storage** for any security types and any tick types
- ✓ In-memory database for **intraday ticks**
- ✓ Complex Event Processing (**CEP**) engine for **real time analysis** & **signal generation**
- ✓ **GUI** with sophisticated extendable **analytics** that can be integrated with **R** & **MatLab**
- ✓ **API** (C++, C#, Java, Perl)

What is OneTick: GUI Analytics

Query Example:
Bollinger Bands
Buy/Sell Signals



Nested query for Bollinger Bands calculations

NOTE: One of the nodes can be **R** Event Processor calling **R** functions

What is OneTick: GUI Analytics

Query Example:
Bollinger Bands Buy/Sell Signals

Symbol	Time	AVERAGE.AVE	STDDEV.STDEV	PRICE	UBAND	LBAND	signal	
O::MSFT.	2010/03/01 15:25:45.065	28.580000000000	0.000000000000	28.580000000000	28.580000000000	28.580000000000	0	C
O::MSFT.	2010/03/01 15:25:45.066	28.580000000000	0.000000000000	28.580000000000	28.580000000000	28.580000000000	0	C
O::MSFT.	2010/03/01 15:25:45.075	28.580000000000	0.000000000000	28.580000000000	28.580000000000	28.580000000000	0	C
O::MSFT.	2010/03/01 15:25:45.117	28.580000000000	0.000000000000	28.580000000000	28.580000000000	28.580000000000	0	C
O::MSFT.	2010/03/01 15:25:45.257	28.584000000000	0.008000000000	28.600000000000	28.584080000000	28.583920000000	-1	C
O::MSFT.	2010/03/01 15:25:45.281	28.586666666666	0.009428090415	28.600000000000	28.586760947570	28.586572385762	0	C

NOTE: Query can be called from **R** passing this output to **R** vector

OneTick and Market Data Management

Market Data Challenges…	OneTick Solution For All Security Types Include…
• Increasing volumes • History vs Real Time	• Archives are compressed, distributed, scalable • Intraday in-memory DB+ CEP Real-time + Archives
• Increasing granularity • Changing schemas • Reference data • Master data	• Time series: flexible schema & nanoseconds • Optimized summary DBs for low frequency data • Reference data design/analytics (naming continuity, contract rollovers, calendars, corporate actions)
• Data cleansing	• Normalization rules start the cleansing process • Data can be updated or deleted by queries • Parts of archives & in-memory DB can be updated
• Complexity • Consolidation • Order books	• Built-in order book storage and analytics • Order book data can be joined or merged with any other kind of data within analytical queries • Join data for derivatives and underlyers
• **Reporting & analytics *vs* all of the above**	• Rich GUI and API analytics • DB structure is transparent to queries • Queries for 1,000s of symbols & multiple sources • Integration: **R**, **MatLab**, **SAS**, **C++**, **Java, Perl**

System Integration: OneTick and R

	OneTick allows to:	R allows to:
If you work in R:	**Pre-process data in OT:** ✓ Normalize and clean data ✓ Apply reference data ✓ Aggregate or intervalize irregular time series ✓ JOIN or MERGE trades with order books, news, etc… ✓ Use OT functions to preprocess ✓ Parameterize queries	✓ **Retrieve OT query output in R code** via ODBC/OneTick SQL: • Call tested OT query • Pass parameters • Results go into an R vector • Process results vector ✓ Limit amount of data processed within ODBC and R
If you work in OneTick GUI:	✓ [All of the above] ✓ **Call any number of OT/R functions** ✓ Call separate R functions for the same tick bucket for bucket entry or exit events ✓ Run historical or CEP queries ✓ View query results in GUI grid or chart, or via API	✓ Map OneTick fields to R parameters ✓ Map R output back to OneTick fields for further OneTick query processing **Tip: Take full advantage of both packages.**

ONEMARKETDATA

Integration Method 1:

ONETICK QUERY RESULTS → BACK TO R

ONEMARKETDATA

OneTick Results Back to R: Overview

Prerequisites: OneTick client setup

Connection: OneTick ODBC Driver

Syntaxes: OneTick SQL (based on SQL)

OneTick Query prep steps:

- ✓ Design query graph with OneTick GUI
- ✓ Create query parameters to be passed from **R**
- ✓ Test and save the query

Results: **R** vector

Via OT ODBC + SQL call with parameters

ONEMARKETDATA

OneTick Results Back to R: Overview

P

C

S

O

TIP:
Pack logic and analytics
into OneTick query
to take advantage of OneTick Server,
and
to limit network traffic and
a number of returned ticks

✓

✓ Create query parameters to be passed from **R**

✓ Test and save the query

Results: **R** vector

Via OT ODBC + SQL call with parameters

ONEMARKETDATA

OneTick Results Back to R: Example

Sample Business Case:

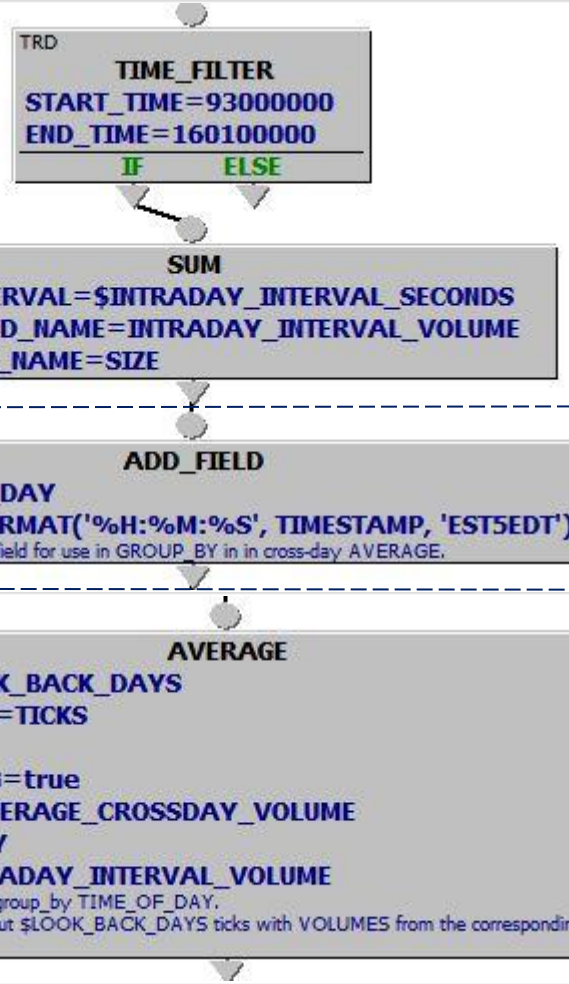
I. **OneTick Query must produce:**
Running liquidity profile (average daily volume over X past days) **grouped by "Time of Day"**.
E.g., average 5-day volume for 9:30 – 10:30, timestamp'ed by the 5th day

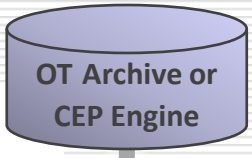
Source trade data: 1 month, 1 symbol

II. **Results** → Back to **R**
for plotting and further processing

OneTick Results Back to R: Design OT Query

OT GUI Query Graph Design





Tick Processing

Approach & OT Steps

- Retrieve trades, filter
- Calculate **VOLUMEs** for all INTRADAY NTERVALS (in seconds)
- Add TIME_OF_DAY fields to be used for grouping
- Calculate **running AVERAGE**
  - for each TIME_OF_DAY
  - over LOOK_BACK_DAYS

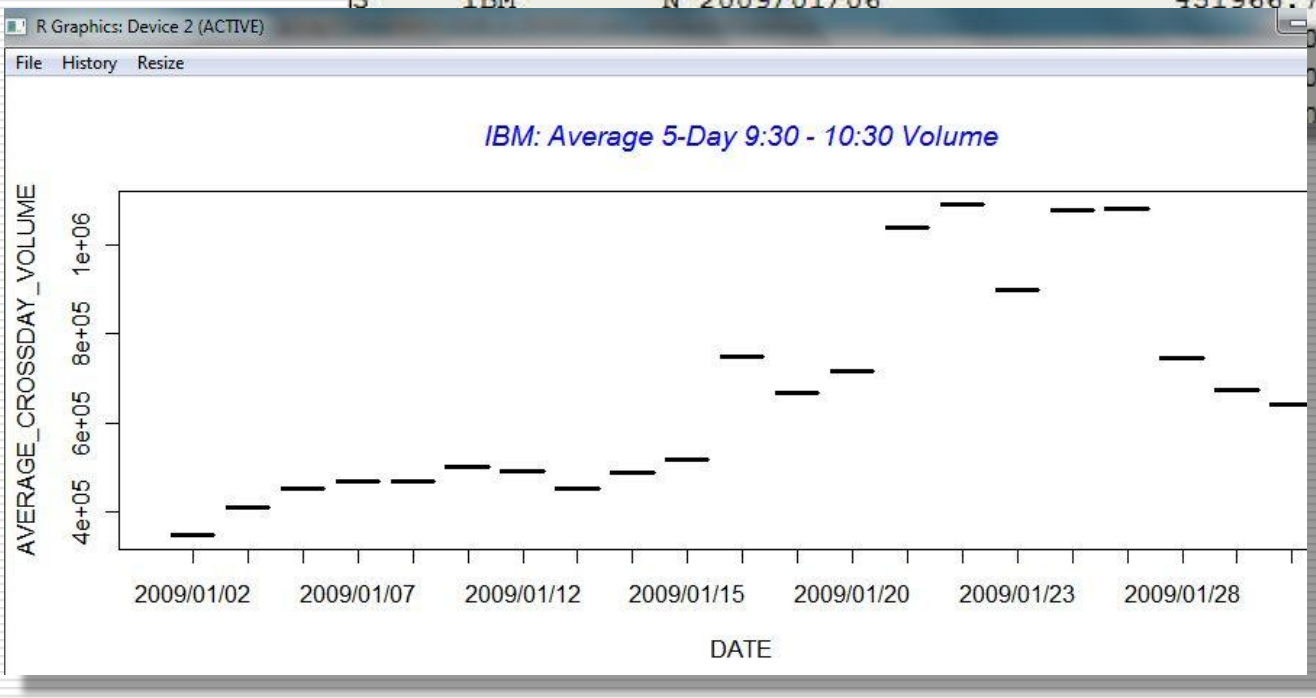
  for **LIQUIDITY PROFILE for each day in query date range**
- **Test** query in GUI

OneTick Results Back to R: Results 1

- ✓ Pass SQL
- ✓ Get query results for 1 month, 9:30 - 10:30 only
- ✓ Plot 5-day volume average by day

REvolution R Enterprise Console

```
> # Print out selected columns:
> results[, c("SYMBOL", "EXCHANGE", "DATE", "AVERAGE_CROSSDAY_VOLUME")]
  SYMBOL EXCHANGE       DATE AVERAGE_CROSSDAY_VOLUME
1    IBM        N 2009/01/02                346800.0
2    IBM        N 2009/01/05                411000.0
3    IBM        N 2009/01/06                451966.7
```

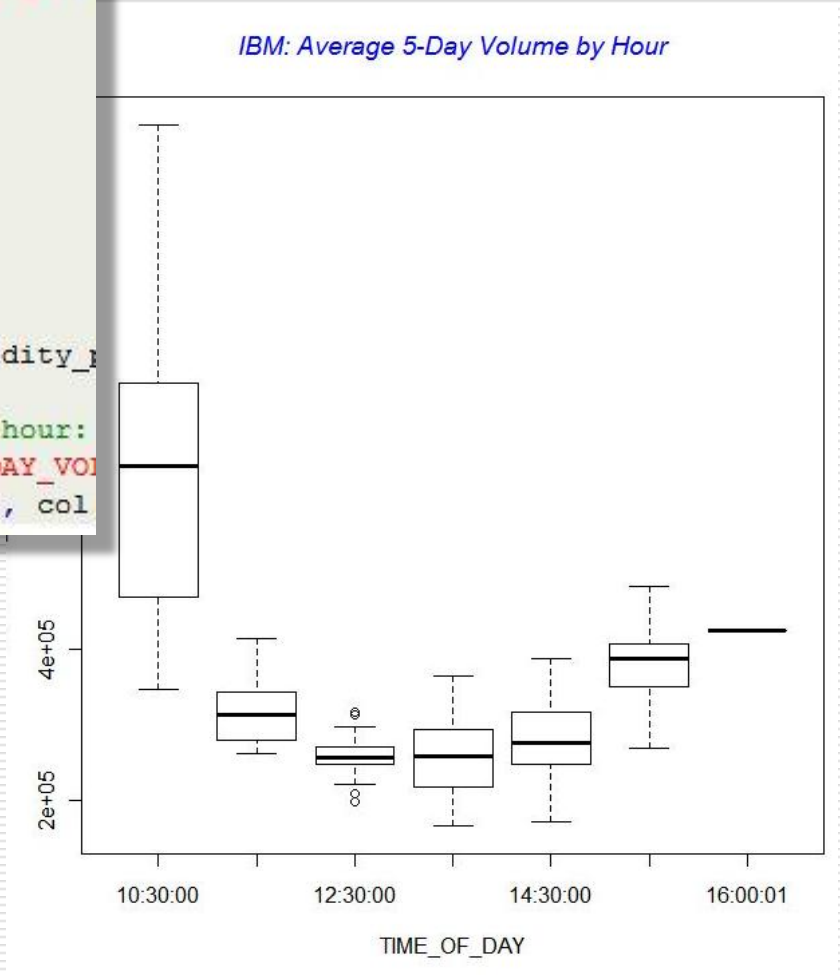


OneTick Results Back to R: Results 2

REvolution R Enterprise Console

```
> # Create a new chart:
> sql<-omdBuildSQLQuery("sample_liquidity_profile_R",
+   "2009-01-02 09:30:00",
+   "2009-01-30 16:00:01",
+   "EST5EDT",
+   DB="TAQ",
+   SYMBOL="IBM.N",
+   INTRADAY_INTERVAL_SECONDS=3600,
+   LOOK_BACK_DAYS=5,
+   TIME_OF_DAY='ALL'
+ ); sql
[1] "SELECT *  FROM OMD.OTQ_FILES.\"sample_liquidity_
> results<-sqlQuery(channel, sql);
> # View behavior of crossday volume average by hour:
> plot(results[, c("TIME_OF_DAY","AVERAGE_CROSSDAY_VO
> title(main="IBM: Average 5-Day Volume by Hour", col
```

- ✓ Pass SQL
- ✓ Get query results for 1 month
- ✓ Plot 5-day volume average values by TIME_OF_DAY



ONEMARKETDATA

Integration Method 2:

ONETICK QUERY ↔ R FUNCTIONS

ONEMARKETDATA

R Functions in OneTick Queries: Overview

Prerequisites: **R** or **REvolution R** installation on the OneTick server (depends on OS)

Connection: Using standard **R** DLL

Syntaxes: OneTick GUI and **R** expressions

Input: OneTick Archive or real-time tick timeseries from a single or multiple data sources

Output: OneTick query results (timeseries defined by the query design)

Query Types: Historical or Continuous CEP

ONEMARKETDATA

R Functions in OneTick: Example 1

Business Case:

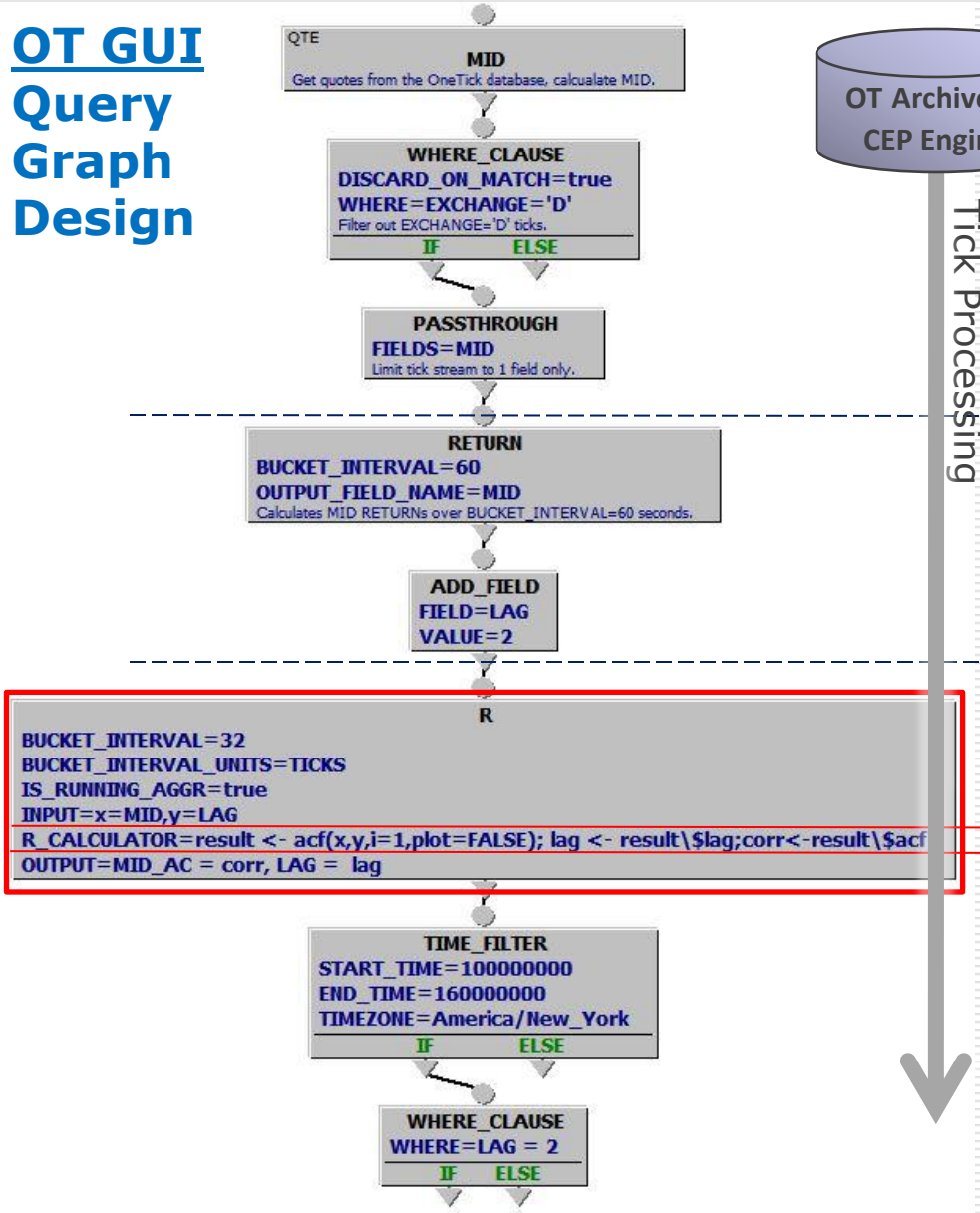
Find serial **autocorrelation**
of 30 one-minute returns with lag of 2

R Function:

acf, an autocorrelation function

R Functions in OneTick: Design Query

OT GUI Query Graph Design



Approach & OT Steps

- Retrieve quote ticks for any number of symbols
- Add tick by tick MID calculation
- Filter ticks

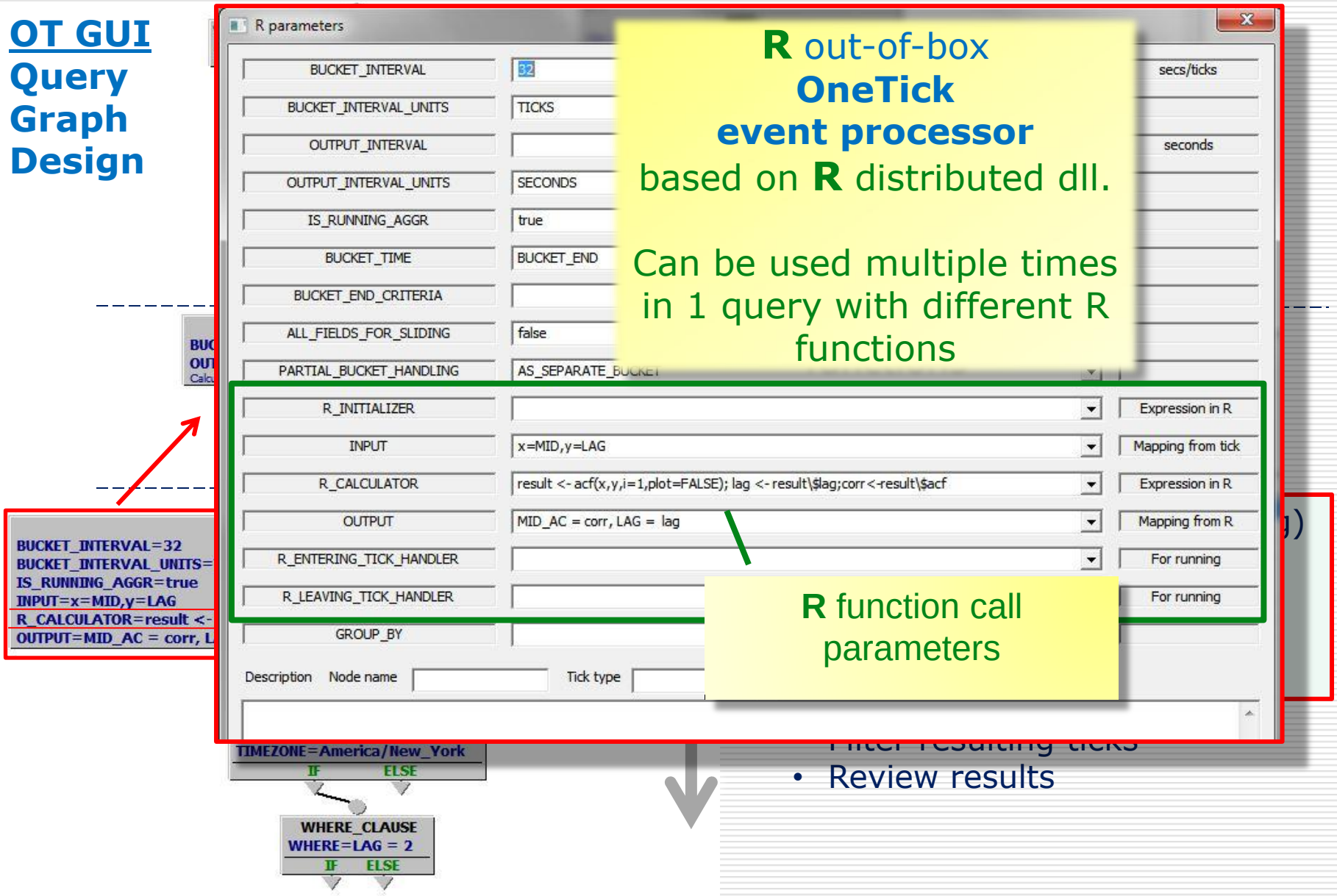
- Aggregate ticks every 60 seconds
- Calculate **RETURN(MID)**
- Add LAG field to each tick

- Create running (a.k.a. sliding) aggregation of 32 ticks
- Call **R** function **acf(...)** for each sliding group
- Pass values of MID and LAG

- Filter resulting ticks
- Review results

R Functions in OneTick: R Processor

OT GUI
Query Graph Design



R out-of-box
OneTick
event processor
based on **R** distributed dll.

Can be used multiple times in 1 query with different R functions

R function call parameters

R Functions in OneTick: Query Output

View OT Query results in a GUI grid or chart

Index	Symbol	Time	LAG	MID_AC
1	FULL_DEMO_L1::CSCO	2006/06/01 10:00:00.000	2.000000000000	-0.254566084117
2	FULL_DEMO_L1::CSCO	2006/06/01 10:01:00.000	2.000000000000	-0.246525291954
3	FULL_DEMO_L1::CSCO	2006/06/01 10:02:00.000	2.000000000000	-0.2427
4	FULL_DEMO_L1::CSCO	2006/06/01 10:03:00.000	2.000000000000	-0.2470
5	FULL_DEMO_L1::CSCO	2006/06/01 10:04:00.000	2.000000000000	-0.2540
6	FULL_DEMO_L1::CSCO	2006/06/01 10:05:00.000	2.000000000000	-0.2534
7	FULL_DEMO_L1::CSCO	2006/06/01 10:06:00.000	2.000000000000	-0.2434
8	FULL_DEMO_L1::CSCO	2006/06/01 10:07:00.000	2.000000000000	-0.2362
9	FULL_DEMO_L1::CSCO	2006/06/01 10:08:00.000	2.000000000000	-0.2339

Date and time | Start | End | Timezone

MM/DD/YYYY 06/01/2006 ... 06/01/2006 ... New York Switch Dataseries

HH:MM:SS.mmm 09:30:00.000 16:30:00.000 Apply times daily Continuous query

Start expression

Or pass query results into API wrapper application : C++, C#, Java, Perl, or a command line query

R Functions in OneTick: R Parameters

2 subsets of parameters that work together:

OneTick aggregation	R function specifications
BUCKET_INTERVAL, UNITS and optional **GROUP_BY** to aggregate ticks into buckets	**R_INITIALIZER** to specify one time only initial command
OUTPUT_INTERVAL and **UNITS** to define frequency of output for running calculations	**INPUT, OUTPUT, R_CALCULATOR** to map tick fields to R variables and specify R command
IS_RUNNING = true/false for running (*a.k.a.* sliding) calculations **Other parameters** for additional flexibility with aggregation	**R_ENTERING/LEAVING_ TICK_HANDLER** to call different functions for BUCKET_INTERVAL **entry** and **exit** events in RUNNING aggregations

Result:
R functions in **OneTick** queries at **any level of granularity,**
with **OneTick** processing and aggregating **large datasets,**
for **any number of symbols**, in **historical** or **real time** queries.

ONEMARKETDATA

Q&A

Contact:

Dr. Maria Belianina

maria.belianina@onetick.com

support@onetick.com