Value At Risk (VaR)
Aggregation and reporting using Spark SQL

Deeper and quicker insight using a High Definition Data Warehouse

- By Deenar Toraskar
  - Co-founder
  - ThinkReactive
Me and Spark

- Hands on
- Many years and many roles at an investment bank
- 20 years of JVM experience
- 10 years+ of risk expertise
- 5 year of big data Hadoop, Hive, Spark, Scala
- Co-Founder of Think Reactive - Risk analytics solutions
- FRTB Market and Credit Risk, Stress Testing, IFRS9 solutions
Background

- Increased Capital Requirements
- More capital charges
- Capital cost is now a large part of PnL
- Capital charges are complex
- Capital charges are portfolio level
- More scenarios are required
Increased Regulatory Measures

- VaR (general + specific)
- Stressed VaR
- CRM
- Floor
- Standardized Charge
- Incremental Risk Charge
- Hypothetical Backtesting
- CVA
- WWR (Wrong Way Risk)
- Stressed EPE
- Liquidity Coverage Ratios
- Net Stable Funding Ratio
- Portfolio Stress Testing
Business Impact

- Regulatory capital costs are now a significant part of trading profit and loss.
- The emphasis on risk management is changing and growing.
- Risk functions are now integral part of trading and risk has now moved from a back-office to a corner-office function.
- Exposures are monitored proactively and risk measures calculations are calculated continuously throughout the trading day.
- In addition regulatory risk impact is now a critical part of pre-trade decision making.
Technology Impact - Perfect Storm

- **Volumes** - More measures, more scenarios, more granular, portfolio level calculations
- **Variety** - More frequent, more timely, near-real time view required
- **Variety** - Complex measures, trade level, more reference and market data required, structured products
- **Veracity** - BCBS lays down general principles for management of risk data sets such as completeness, traceability, accuracy, validation, reconciliation and integrity
Agenda

- VaR reporting and Aggregation on Spark
  - What is Value at Risk (VaR)?
  - How do you calculate VaR?
  - Challenges in VaR reporting and aggregation
  - Spark SQL solution
- High Definition Data Warehouse + DSL => Empowered Users
- Questions + Demo (time permitting)
What is Value At Risk (VaR)

- Predicted maximum loss (or worst loss) over a target horizon within a given confidence interval.
- $1000 GOOG shares => 1 day Value at Risk at a 95% confidence level is $22. (99% - $31)
- i.e. You would lose less than $22, 95 days out of 100 over one trading day.
- More volatile the portfolio larger the value at risk, VaR (95%, $1000 of TWTR) =
What is VaR used for?

- Widely adopted risk measure since stock market crash of 1987
- Risk Management
- Broker Margin Calculations
- Regulatory Reporting
- Capital Charges
- Pre-Trade Decision
How do you calculate VaR?

- VaR of risk of a single asset viz. $1,000 worth of GOOG shares
- Find price history GOOG going back in time
- Get Prices, convert to 1 day returns and PnL https://uk.finance.yahoo.com/q/hp?s=GOOG
- Sort the PnLs
- Take the 5% worst PnL (PERCENTILE in excel) -2.14% VaR (95%) = -22, VaR (99%) = -36
- Historical Simulation VaR
## Daily Returns

<table>
<thead>
<tr>
<th>Date</th>
<th>Closing Price</th>
<th>Daily Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/30/2015</td>
<td>537.34003</td>
<td>-2.138%</td>
</tr>
<tr>
<td>4/29/2015</td>
<td>549.08002</td>
<td>-0.831%</td>
</tr>
<tr>
<td>4/28/2015</td>
<td>553.67999</td>
<td>-0.304%</td>
</tr>
<tr>
<td>4/27/2015</td>
<td>555.37</td>
<td>-1.715%</td>
</tr>
<tr>
<td>4/24/2015</td>
<td>565.06</td>
<td>3.302%</td>
</tr>
<tr>
<td>4/23/2015</td>
<td>547</td>
<td>1.416%</td>
</tr>
<tr>
<td>4/22/2015</td>
<td>539.36499</td>
<td>1.010%</td>
</tr>
<tr>
<td>4/21/2015</td>
<td>533.96997</td>
<td>-0.263%</td>
</tr>
<tr>
<td>4/20/2015</td>
<td>535.38</td>
<td>2.162%</td>
</tr>
<tr>
<td>4/17/2015</td>
<td>524.04999</td>
<td>-1.827%</td>
</tr>
<tr>
<td>4/16/2015</td>
<td>533.79999</td>
<td>0.238%</td>
</tr>
<tr>
<td>4/15/2015</td>
<td>532.53003</td>
<td>0.403%</td>
</tr>
<tr>
<td>4/14/2015</td>
<td>530.39001</td>
<td>-1.628%</td>
</tr>
<tr>
<td>4/13/2015</td>
<td>539.16998</td>
<td>-0.156%</td>
</tr>
<tr>
<td>4/10/2015</td>
<td>540.01001</td>
<td>-0.142%</td>
</tr>
<tr>
<td>4/9/2015</td>
<td>540.78003</td>
<td>-0.153%</td>
</tr>
<tr>
<td>4/8/2015</td>
<td>541.60999</td>
<td>0.855%</td>
</tr>
<tr>
<td>4/7/2015</td>
<td>537.02002</td>
<td>0.048%</td>
</tr>
</tbody>
</table>

Daily Return = 
(Today’s Price - Yesterday’s Price)/Yesterday’s Price
### Sorted - Worst Daily Losses

<table>
<thead>
<tr>
<th>Date</th>
<th>Daily Return</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/16/2014</td>
<td>-3.583%</td>
<td>-$35.83</td>
</tr>
<tr>
<td>1/27/2015</td>
<td>-3.098%</td>
<td>-$30.98</td>
</tr>
<tr>
<td>10/10/2014</td>
<td>-2.922%</td>
<td>-$29.22</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>-2.693%</td>
<td>-$26.93</td>
</tr>
<tr>
<td>10/17/2014</td>
<td>-2.543%</td>
<td>-$25.43</td>
</tr>
<tr>
<td>3/10/2015</td>
<td>-2.433%</td>
<td>-$24.33</td>
</tr>
<tr>
<td>10/7/2014</td>
<td>-2.357%</td>
<td>-$23.57</td>
</tr>
<tr>
<td>1/6/2015</td>
<td>-2.318%</td>
<td>-$23.18</td>
</tr>
<tr>
<td>12/5/2014</td>
<td>-2.243%</td>
<td>-$22.43</td>
</tr>
<tr>
<td>4/30/2015</td>
<td>-2.138%</td>
<td>-$21.38</td>
</tr>
<tr>
<td>1/5/2015</td>
<td>-2.085%</td>
<td>-$20.85</td>
</tr>
<tr>
<td>10/13/2014</td>
<td>-2.072%</td>
<td>-$20.72</td>
</tr>
<tr>
<td>10/9/2014</td>
<td>-2.030%</td>
<td>-$20.30</td>
</tr>
<tr>
<td>3/25/2015</td>
<td>-2.000%</td>
<td>-$20.00</td>
</tr>
<tr>
<td>12/12/2014</td>
<td>-1.832%</td>
<td>-$18.32</td>
</tr>
<tr>
<td>4/17/2015</td>
<td>-1.827%</td>
<td>-$18.27</td>
</tr>
<tr>
<td>4/27/2015</td>
<td>-1.715%</td>
<td>-$17.15</td>
</tr>
<tr>
<td>1/28/2015</td>
<td>-1.664%</td>
<td>-$16.64</td>
</tr>
</tbody>
</table>

**Value at Risk (99%), 1% worst return (2nd worst out of 200)**

**Value at Risk 95% confidence, 5% worst return**

![Daily GOOG Returns](image)
def valueAtRisk(pnls : Seq[Double], confidence: Double): Double = {
    val size = pnls.size           /* find number of scenarios */
    val sortedVector = pnls.sorted /* sort returns */
    val indexR = (size * ((100 - confidence) / 100)) - 1 /* percentile */
    val upper = math.ceil(indexR).toInt.max(0).min(size - 1)
    val lower = math.floor(indexR).toInt.max(0).min(size - 1)
    if (lower == upper)
        sortedVector.apply(upper)
    else    ((upper - indexR) * sortedVector(lower)) + ((indexR - lower) * sortedVector(upper)) /* interpolate if percentile between scenarios */
}
Varied VaR reporting requirements

- VaR by counterparty - for margining
- VaR by trader - Trader P&L
- VaR by Desk - Desk head
- VaR by Legal Entity - Finance/Regulator
- ....
- What days drove VaR?
Value At Risk (VaR) of a portfolio

- VaR is not simply additive
- VAR of a portfolio containing assets A and B does not equal the sum of VAR of asset A and VAR of asset B.

\[
\text{VaR} (\$500 \text{ GOOG} + \$500 \text{ BERK}) \\
\Leftrightarrow \\
\text{VaR}($500 \text{ GOOG}) + \text{VaR}($500 \text{ BERK})
\]
Value At Risk (VaR) of a portfolio

- Add individual position PnLs for each day to get daily portfolio PnL
- Then sort portfolio PnLs
- Take percentile to calculate Portfolio VaR
# Portfolio Returns

<table>
<thead>
<tr>
<th>Date</th>
<th>GOOG</th>
<th>GOOG Return</th>
<th>BERK</th>
<th>BERK Return</th>
<th>Portfolio PnL</th>
<th>Total</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/30/2015</td>
<td>537.34003</td>
<td>-2.138%</td>
<td>141.21001</td>
<td>-0.843%</td>
<td>-14.90</td>
<td>-1.49%</td>
<td></td>
</tr>
<tr>
<td>4/29/2015</td>
<td>549.08002</td>
<td>-0.831%</td>
<td>142.41</td>
<td>-0.077%</td>
<td>-4.54</td>
<td>-0.45%</td>
<td></td>
</tr>
<tr>
<td>4/28/2015</td>
<td>553.67999</td>
<td>-0.304%</td>
<td>142.52</td>
<td>1.014%</td>
<td>3.55</td>
<td>0.35%</td>
<td></td>
</tr>
<tr>
<td>4/27/2015</td>
<td>555.37</td>
<td>-1.715%</td>
<td>141.09</td>
<td>-0.704%</td>
<td>-12.09</td>
<td>-1.21%</td>
<td></td>
</tr>
<tr>
<td>4/24/2015</td>
<td>565.06</td>
<td>3.302%</td>
<td>142.09</td>
<td>-0.239%</td>
<td>15.31</td>
<td>1.53%</td>
<td></td>
</tr>
<tr>
<td>4/23/2015</td>
<td>547</td>
<td>1.416%</td>
<td>142.42999</td>
<td>0.310%</td>
<td>8.63</td>
<td>0.86%</td>
<td></td>
</tr>
<tr>
<td>4/22/2015</td>
<td>539.36499</td>
<td>1.010%</td>
<td>141.99001</td>
<td>0.602%</td>
<td>8.06</td>
<td>0.81%</td>
<td></td>
</tr>
<tr>
<td>4/21/2015</td>
<td>533.96997</td>
<td>-0.263%</td>
<td>141.14</td>
<td>-0.669%</td>
<td>-4.66</td>
<td>-0.47%</td>
<td></td>
</tr>
<tr>
<td>4/20/2015</td>
<td>535.38</td>
<td>2.162%</td>
<td>142.09</td>
<td>0.988%</td>
<td>15.75</td>
<td>1.57%</td>
<td></td>
</tr>
<tr>
<td>4/17/2015</td>
<td>524.04999</td>
<td>-1.827%</td>
<td>140.7</td>
<td>-0.985%</td>
<td>-14.06</td>
<td>-1.41%</td>
<td></td>
</tr>
</tbody>
</table>

Portfolio PnL = Asset A PnL + Asset B PnL
## Sorted - Worst Daily Returns

<table>
<thead>
<tr>
<th>Date</th>
<th>Return</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>7/31/2014</td>
<td>-2.251%</td>
<td>-$22.51</td>
</tr>
<tr>
<td>1/27/2015</td>
<td>-2.234%</td>
<td>-$22.34</td>
</tr>
<tr>
<td>3/10/2015</td>
<td>-2.222%</td>
<td>-$22.22</td>
</tr>
<tr>
<td>12/12/2014</td>
<td>-2.138%</td>
<td>-$21.38</td>
</tr>
<tr>
<td>9/25/2014</td>
<td>-2.121%</td>
<td>-$21.21</td>
</tr>
<tr>
<td>12/16/2014</td>
<td>-2.079%</td>
<td>-$20.79</td>
</tr>
<tr>
<td>10/7/2014</td>
<td>-2.055%</td>
<td>-$20.55</td>
</tr>
<tr>
<td>10/9/2014</td>
<td>-2.033%</td>
<td>-$20.33</td>
</tr>
<tr>
<td>1/28/2015</td>
<td>-1.804%</td>
<td>-$18.04</td>
</tr>
<tr>
<td>1/5/2015</td>
<td>-1.770%</td>
<td>-$17.70</td>
</tr>
<tr>
<td>10/10/2014</td>
<td>-1.581%</td>
<td>-$15.81</td>
</tr>
<tr>
<td>4/30/2015</td>
<td>-1.490%</td>
<td>-$14.90</td>
</tr>
<tr>
<td>4/17/2015</td>
<td>-1.406%</td>
<td>-$14.06</td>
</tr>
<tr>
<td>7/17/2014</td>
<td>-1.387%</td>
<td>-$13.87</td>
</tr>
</tbody>
</table>

Value at Risk (99%), 1% worst return

Value at Risk 95% confidence, 5% worst return
CREATE TABLE positionsVaR (book INT, trader String, clientAccount String, product String riskFactor String, riskType String, var Double)

Select book, sum(VaR) from positionsVaR group by book

(* does not work *)

- Pre-aggregate all VaR values by certain dimensions
- In turn users lose the flexibility to analyse risk metrics in detail. Dependent on IT or batch jobs for further analysis.
Traditional Data Warehouse

- Shallow reporting schema, limited analytical capabilities
- Support standard slice and dice and aggregation functions used for reporting
- No support for non-linear or semi structured data, such as vectors or json objects
- Pre-aggregation of data used for performance and presenting analytics results from engines
- New analytics/aggregation level require new views and changes
VaR Aggregation and Reporting
Spark SQL solution
VaR Aggregation with Spark SQL

1) CREATE TABLE positions

(book INT, trader String, clientAccount String, product String riskFactor String, riskType String, nominal Double, pnls Array<Double>)

1) sqlContext.udf.register("valueAtRisk", valueAtRisk _ )
sqlContext.sql("create temporary function arraySum as 'com.thinkreactive.udf.GenericUDAFArraySum';")

1) select book, valueAtRisk(arraySum(pnls), 95.0) as VaR from positions group by book

High definition view of data + DSL
/* VaR by client */
$ select clientAccount, valueAtRisk(arraySum(pnls), 99.0)
from positions group by clientAccount"

/* What If I sell GOOG */
$ select clientAccount, valueAtRisk(arraySum(pnls), 99.0)
from positions where riskFactor <> 'GOOG' group by
clientAccount

/* Change confidence level */
$select clientAccount, valueAtRisk(arraySum(pnls), 95.0)
from positions group by clientAccount
• Enrich your data warehouse by storing a higher definition view of your data using complex type support and provide a DSL using User defined functions

• pnl Array<Double> + valueAtRisk( .. )

• Entire risk profile of positions in the data model allows users to ask any questions using the reporting layer
• SQL - lingua franca of data analysis
• Custom UDFs
• Hundreds of standard Hive and community contributed UDFs, from likes of Facebook, Brickhouse, etc.
• Building blocks, Chain functions
• UDFs in Spark - concise, easily testable
• Write once - use everywhere - Streaming, batch jobs, REST services, adhoc queries (explore use cases), Machine Learning

SQL + UDFs = DSL
More Take Aways

- Hadoop and No-SQL databases provide scalable, resilient storage.
- Avro, Thrift or Protocol Buffers, you can store your data using rich, evolvable schemas.
- Kafka provides fast, scalable, durable and distributed messaging infrastructure.
- Lambda Architecture defines a consistent approach to choosing and wiring Big Data technologies together.
• Hive Complex Types - Maps, Arrays and Struts to represent your model
• Don't ETL, write custom Data Source/Serde
• Power of SQL and all other Hadoop tools are at your finger tips
• Use standard visualisation tools - Tableau, ZoomDate, Qlik, Microstrategy, any JDBC/ODBC compliant tool
• Empowered users – leaving engineering team to work on more things
• Changes driven by BCBS - FRTB (aka Basel 4)
• VaR -> Expected Shortfall
• Scaled Expected Shortfall (Liquidity factor)
• Desk level treatment
• Change time horizon (10 day VaR)
• Change confidence level
• Complete Transparency - find VaR drivers
/* alternative risk measure to VaR that is more sensitive to the shape of the loss distribution in the tail of the distribution

a.k.a. **conditional value at risk (CVaR)**, **average value at risk (AVaR)**, and **expected tail loss (ETL)**. */

def expectedShortfall(pnls : Seq[Double], percentile: Double): Double = {
  val size = pnls.size
  val sortedVector = pnls.sorted
  val indexR = (size * ((100 - percentile) / 100)) - 1
  val upper = math.ceil(indexR).toInt.max(0).min(size - 1)
  val lower = math.floor(indexR).toInt.max(0).min(size - 1)
  if (lower == upper)
    sortedVector.take(lower)/lower
  else
    (sortedVector.take(lower) + ((upper - indexR) * sortedVector(lower)) + ((indexR - lower) * sortedVector(upper)))/indexR}
Polygot Persistence

- Positions from HDFS
- TimeSeries from Cassandra
- Hierarchy, Liquidity horizons and other reference data from RDBMS
val url = "jdbc:oracle:thin:@data.thinkreactive.co.uk:1922:DATA"
val jdbcDF = sqlContext.load("jdbc", Map(  "url" -> url,
                               "dbtable" -> "Hierarchy"))
jdbcDF.registerTempTable("hierarchy")

/* VaR by business hierarchy */
select businessArea, desk, trader. valueAtRisk(pnls, 95.0) from positions left outer join hierarchy on (positions.bookId= hierarchy.bookId) group by businessArea, desk, trader
Thank You

- Rate my session
- Feel free to reach out
- Ready to go risk analytics