

Apache Zeppelin

A Notebook Interface to our Big Data

Bernard Lee Kok Bang

Apache Zeppelin

- primarily used as a quick way to experiment with Apache Spark script
- visualize our big data in an interactive manner
- has plug-ins for other components in Hadoop cluster, like HBase, Cassandra...
- a tool for doing data science on HDFS cluster

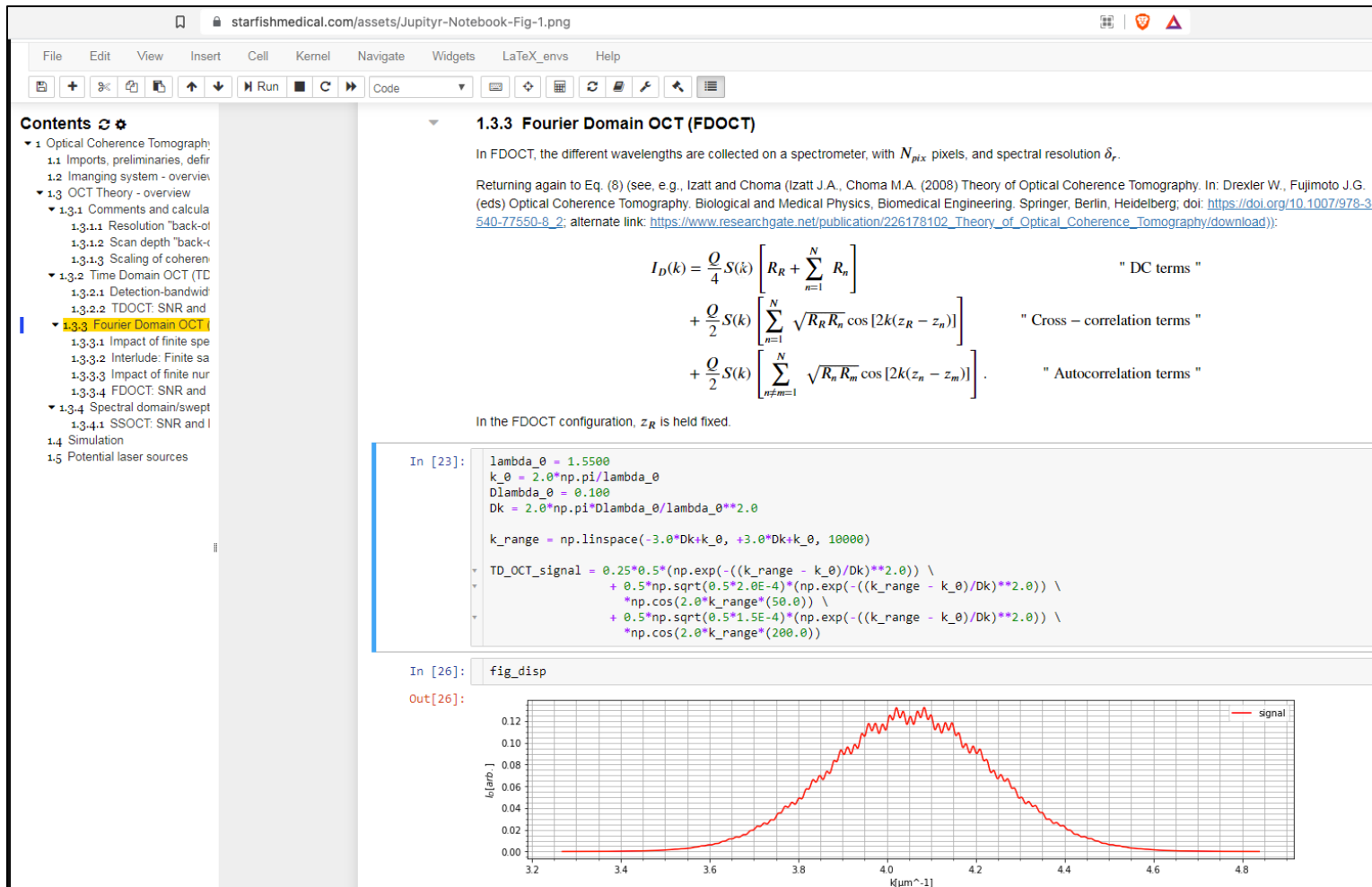


Apache Zeppelin

What is Zeppelin

- It's a *notebook*
- Let us interactively run scripts / code against the data
- interleave with nicely formatted notes
- Share notebooks with others on the cluster
- Speed up development cycle
- Just like Jupyter Notebook (Python) or R Notebook (R)

Example of Jupyter Notebook







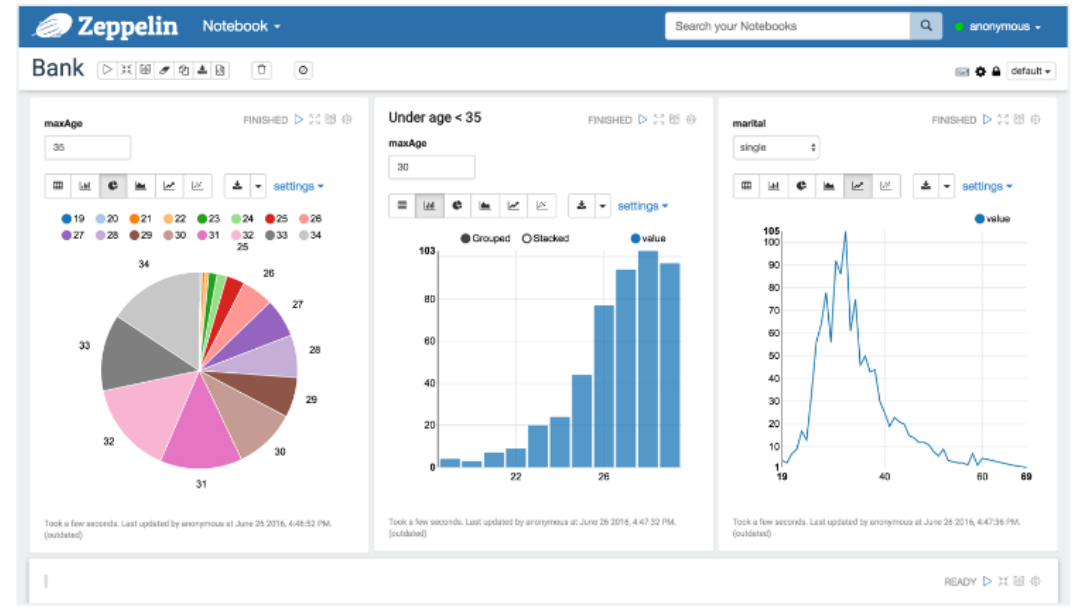
<https://bit.ly/3ugdAV0>

Zeppelin notebook

Multi-purpose Notebook

The Notebook is the place for all your needs

-  Data Ingestion
-  Data Discovery
-  Data Analytics
-  Data Visualization & Collaboration



<https://bit.ly/3bHbWoY>

Apache Spark integration

- Can run Spark code interactively
 - allows easy experimentation and exploration of the big data
- Can execute SQL queries directly against SparkSQL
- Query results may be visualized in charts and graphs
- Make Spark feel more like a data science tool!

instead of writing Spark script in puTTY and run using spark-submit wondering what the output would be

Multiple Language Backend

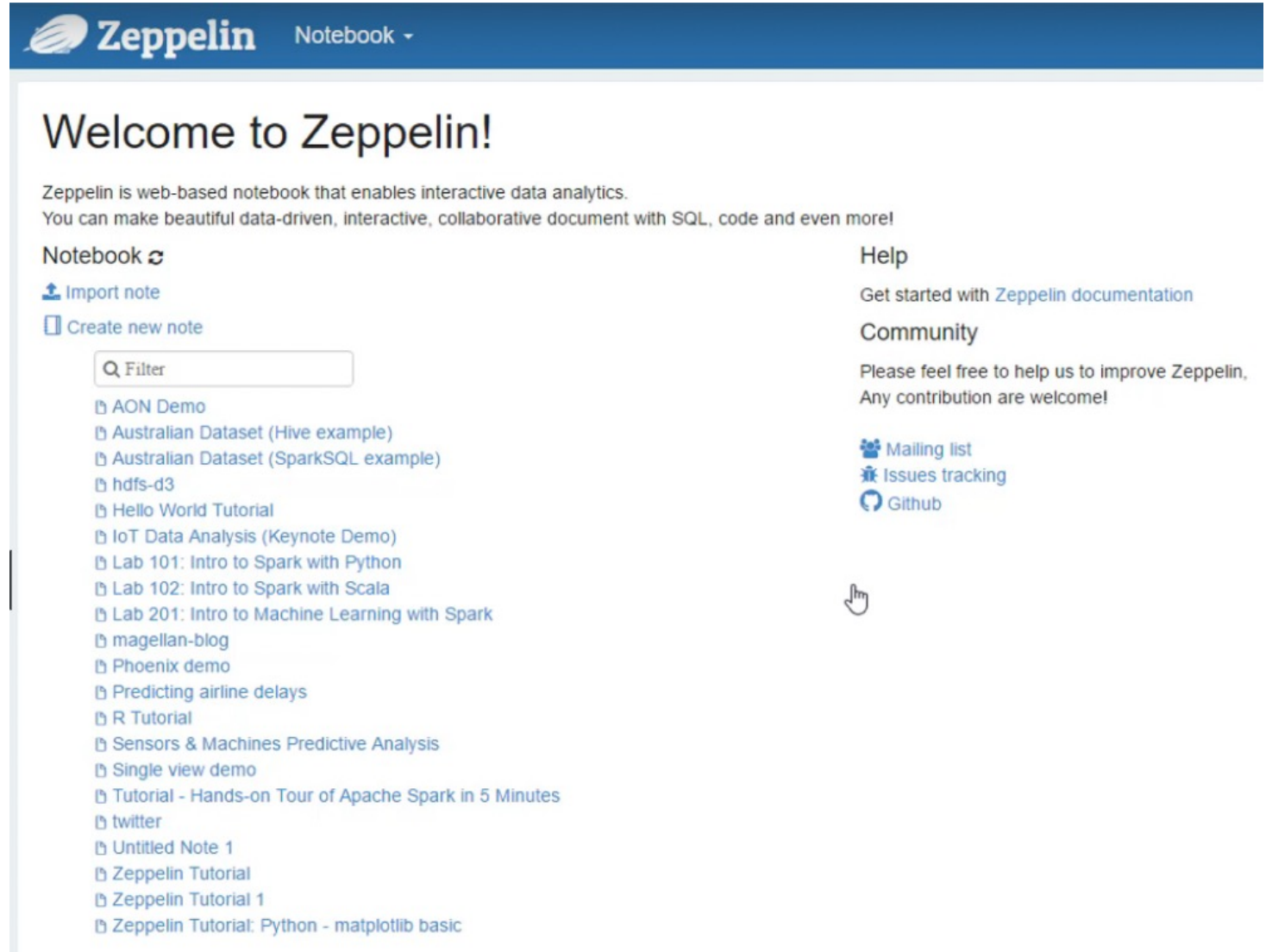
Apache Zeppelin interpreter concept allows any language/data-processing-backend to be plugged into Zeppelin. Currently Apache Zeppelin supports many interpreters such as Apache Spark, Apache Flink, Python, R, JDBC, Markdown and Shell.



<https://bit.ly/3bHbWoY>

To connect Zeppelin

- Open a browser window, connect to the following port number
 - **127.0.0.1:9995**
- Click “**Create new note**” button to start writing a notebook



The screenshot shows the Zeppelin Notebook web interface. At the top, there is a blue header with the Zeppelin logo and the text "Notebook". Below the header, the main content area has a large heading "Welcome to Zeppelin!". Underneath this heading, there is a brief description: "Zeppelin is web-based notebook that enables interactive data analytics. You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!".

Below the description, there are two buttons: "Import note" (with a download icon) and "Create new note" (with a document icon). To the right of these buttons, there is a "Help" section with a link to "Get started with Zeppelin documentation". Below the "Help" section, there is a "Community" section with a message: "Please feel free to help us to improve Zeppelin, Any contribution are welcome!". Under the "Community" section, there are three links: "Mailing list", "Issues tracking", and "Github".

Below the "Create new note" button, there is a search bar labeled "Filter". Below the search bar, there is a list of notebook titles, each preceded by a document icon. The titles are: "AON Demo", "Australian Dataset (Hive example)", "Australian Dataset (SparkSQL example)", "hdfs-d3", "Hello World Tutorial", "IoT Data Analysis (Keynote Demo)", "Lab 101: Intro to Spark with Python", "Lab 102: Intro to Spark with Scala", "Lab 201: Intro to Machine Learning with Spark", "magellan-blog", "Phoenix demo", "Predicting airline delays", "R Tutorial", "Sensors & Machines Predictive Analysis", "Single view demo", "Tutorial - Hands-on Tour of Apache Spark in 5 Minutes", "twitter", "Untitled Note 1", "Zeppelin Tutorial", "Zeppelin Tutorial 1", and "Zeppelin Tutorial: Python - matplotlib basic".

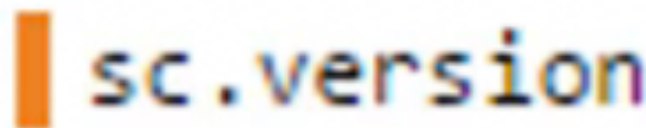
Hands on: Playing with Movielens using Zeppelin

- First, need to prioritize the backend interpreters
- click the *gear button* on top right → *interpreter binding*
- drag *spark* to the top (if it is not), and *md* second

```
%md  
### Let's make sure Spark is working first!  
Let's see what version we're working with.
```

Let's make sure Spark is working first!

Let's see what version we're working with.



Execute shell command in Zeppelin

```
%sh
```

```
wget http://media.sundog-soft.com/hadoop/ml-100k/u.data -O /tmp/u.data  
wget http://media.sundog-soft.com/hadoop/ml-100k/u.item -O /tmp/u.item  
echo "Downloaded!"
```

Copy data to HDFS

```
%sh
```

```
hadoop fs -rm -r -f /tmp/ml-100k
```

```
hadoop fs -mkdir /tmp/ml-100k
```

```
hadoop fs -put /tmp/u.data /tmp/ml-100k/
```

```
hadoop fs -put /tmp/u.item /tmp/ml-100k/
```

Write Scala code (primary interpreter for Spark!!!)

- Extract 2nd and 3rd field from u.data

```
final case class Rating(movieID: Int, rating: Int)

val lines = sc.textFile("hdfs:///tmp/ml-100k/u.data").map(x => {val fields = x.split("\t"); Rating(fields(1).toInt, fields(2).toInt)})
```

- Create Spark DataFrame

```
import sqlContext.implicits._
val ratingsDF = lines.toDF()

ratingsDF.printSchema()
```

Write Scala code [continued...]

- Get top movieIDs

```
val topMovieIDs = ratingsDF.groupBy("movieID").count().orderBy(desc("count")).cache()  
topMovieIDs.show()
```

Write SQL query

- Create a table [convert from dataframe to a table]

```
ratingsDF.registerTempTable("ratings")
```

- Run SQL command

```
%sql  
SELECT * FROM ratings LIMIT 10
```

```
%sql  
SELECT rating, COUNT(*) as count FROM ratings GROUP BY rating
```

Get movie title name

```
final case class Movie(movieID: Int, title: String)

val lines = sc.textFile("hdfs:///tmp/ml-100k/u.item").map(x => {val fields = x.split('|'); Movie(fields(0).toInt, fields(1))})

import sqlContext.implicits._
val moviesDF = lines.toDF()

moviesDF.show()
```

```
moviesDF.registerTempTable("titles")
```

```
%sql
SELECT t.title, count(*) cnt FROM ratings r JOIN titles t ON r.movieID = t.movieID GROUP BY t.title ORDER BY cnt DESC LIMIT 20
```

Remember to share your Zeppelin notebook with Others!!!

Hue – *H*adoop *U*ser *E*xperience 😊

- A close competitor of Hortonworks Data Platform [HDP]
- access Hue here: gethue.com



😊Thanks for tagging along on this
journey for the past one year 😊

Good Luck !!!

&

until we meet again, take care!!!

DEPART
TO SERVE BETTER THY COUNTRY AND THY KIND