

JDemetra+

*an open framework for seasonal
adjustment and time series
methods for official statistics*

ESTP training

0. Outline

- Objectives of JDemetra+ (JD+)
 - General
 - For seasonal adjustment (SA)
- What is really JD+ ?
- Statistical content
- Seasonal adjustment framework
- Final remarks

1.1 General Objectives

- Providing algorithms for the production/analysis of [official] statistics
 - Regular time series (from monthly to yearly)
 - Algorithms for
 - Seasonal adjustment, business cycle analysis
 - Benchmarking, temporal disaggregation
 - Modelling (forecasting, estimation of missing values, outliers detection)
- Reusable modules, compatible with common IT infrastructure
 - Java, R, WEB services...
- Designed for the whole statistical process
 - From research to bulk production (flexible, high-performance)
- Maintainable
 - Open source solution

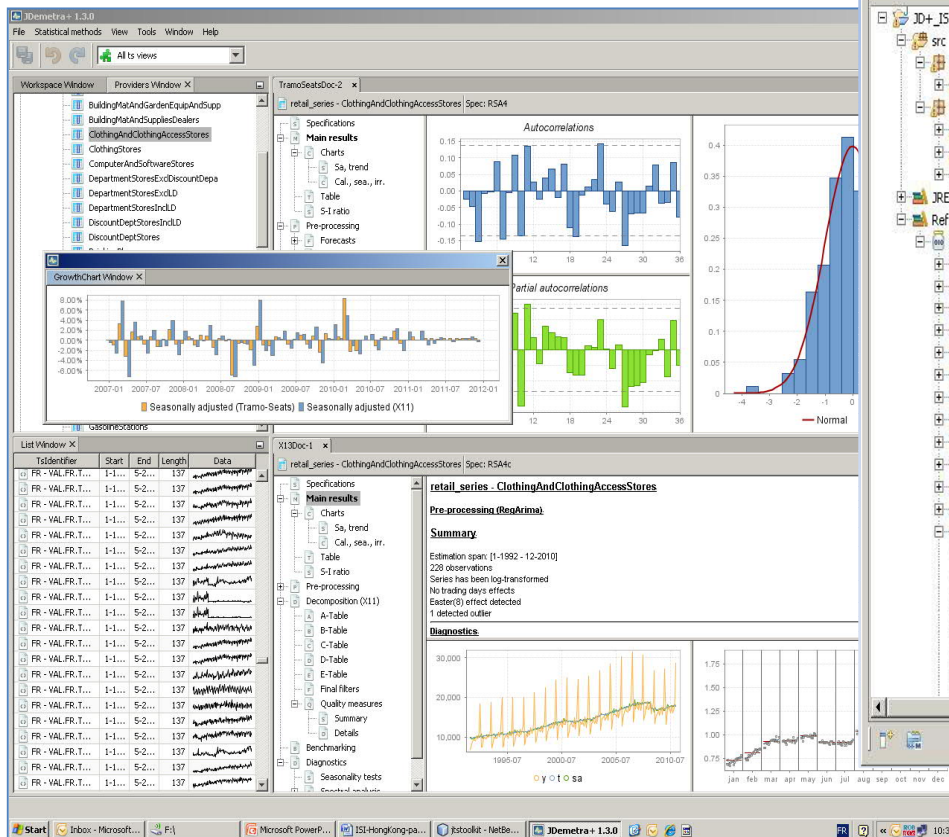
1.2 Objectives for SA

- Java implementation of the leading algorithms
 - Tramo-Seats, X12-ARIMA...
- Flexible design
 - Easier modifications of the core engines
 - Developments of additional tools/algorithms
- Challenge
 - Keeping
 - similar results
 - high performances
 - with
 - flexible (more general) design and algorithms
 - slower technical solution

2.1 What is JD+ (I) ?

Advanced Java toolkit for time series
(SA) processing (IT-teams, researchers)

Rich graphical application (end-users)



The screenshot displays the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The Package Explorer on the left shows the project structure for 'JD+_ISI'. The main editor window displays the code for 'App.java', which defines two static methods: 'series' and 'canonical'. The 'series' method creates a 'TsData' object with a specified frequency and start date. The 'canonical' method creates a 'UcarimaModel' object using the 'SarimaModelBuilder' and 'TrendCycleSelector' classes. The code is written in Java and uses the 'demetra-toolkit' library.

```

public static TsData series(int n) {
    DataBlock data = new DataBlock(n);
    data.randomize();
    TsPeriod start = new TsPeriod(TsFrequency.Monthly, 1980, 0);
    return new TsData(start, data);
}

public static UcarimaModel canonical() {
    SarimaModel sarima = new ec.tstoolkit.sarima.SarimaModelBuilder()
        .createAirlineModel(12, -.6, -.4);
    TrendCycleSelector tsel = new TrendCycleSelector();
    SeasonalSelector ssel = new SeasonalSelector(sarima.getSpecification()
        .getFrequency());

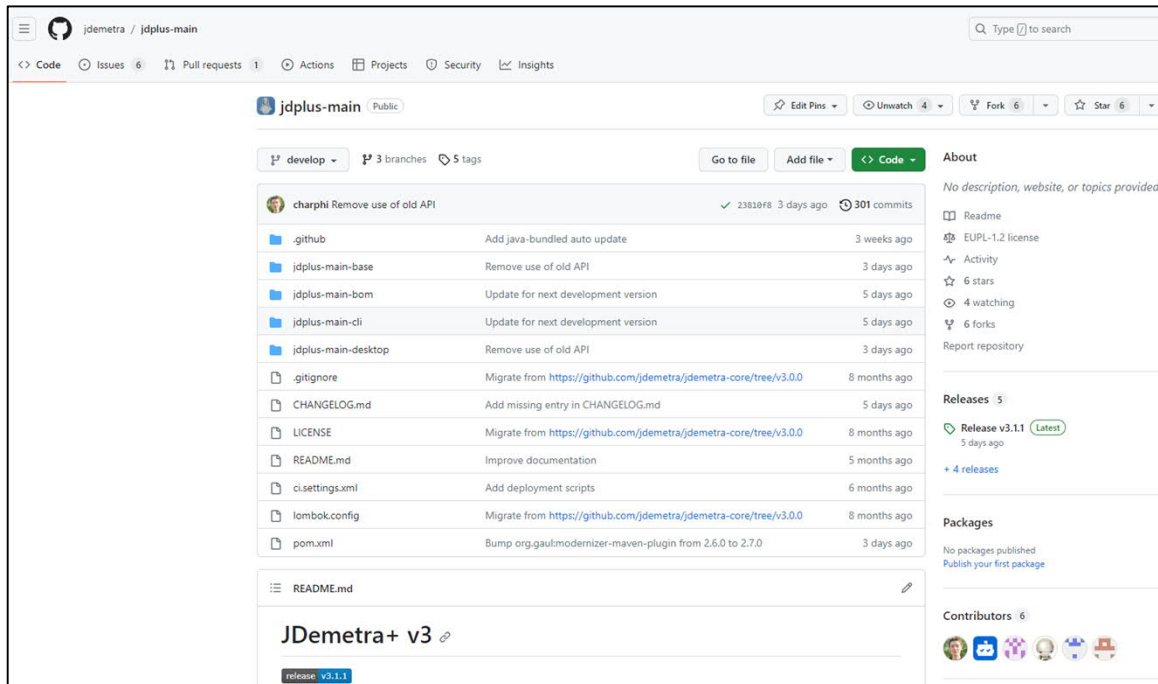
    ModelDecomposer decomposer = new ModelDecomposer();
    decomposer.add(tsel);
    decomposer.add(ssel);

    UcarimaModel ucm = decomposer.decompose(ArimaModel.create(sarima));
    ucm.setVarianceMax(-1);
    ucm.simplify();
    return ucm;
}

public static TsData saMcElroy(UcarimaModel m, TsData ts) {
    McElroyEstimates mc = new McElroyEstimates();
    mc.setUcarimaModel(m);
    mc.setData(ts);
    double[] s=mc.getComponent(1);
    TsData seas=new TsData(ts.getStart(), s,false);
    return ts.minus(seas);
}

```

2.2 What is JD+ ? (II)



Open Source project (EUPL license)

- Supported by Eurostat

• Developers:

- NBB
- Bundesbank
- INSEE
- ...

• Originally based on:

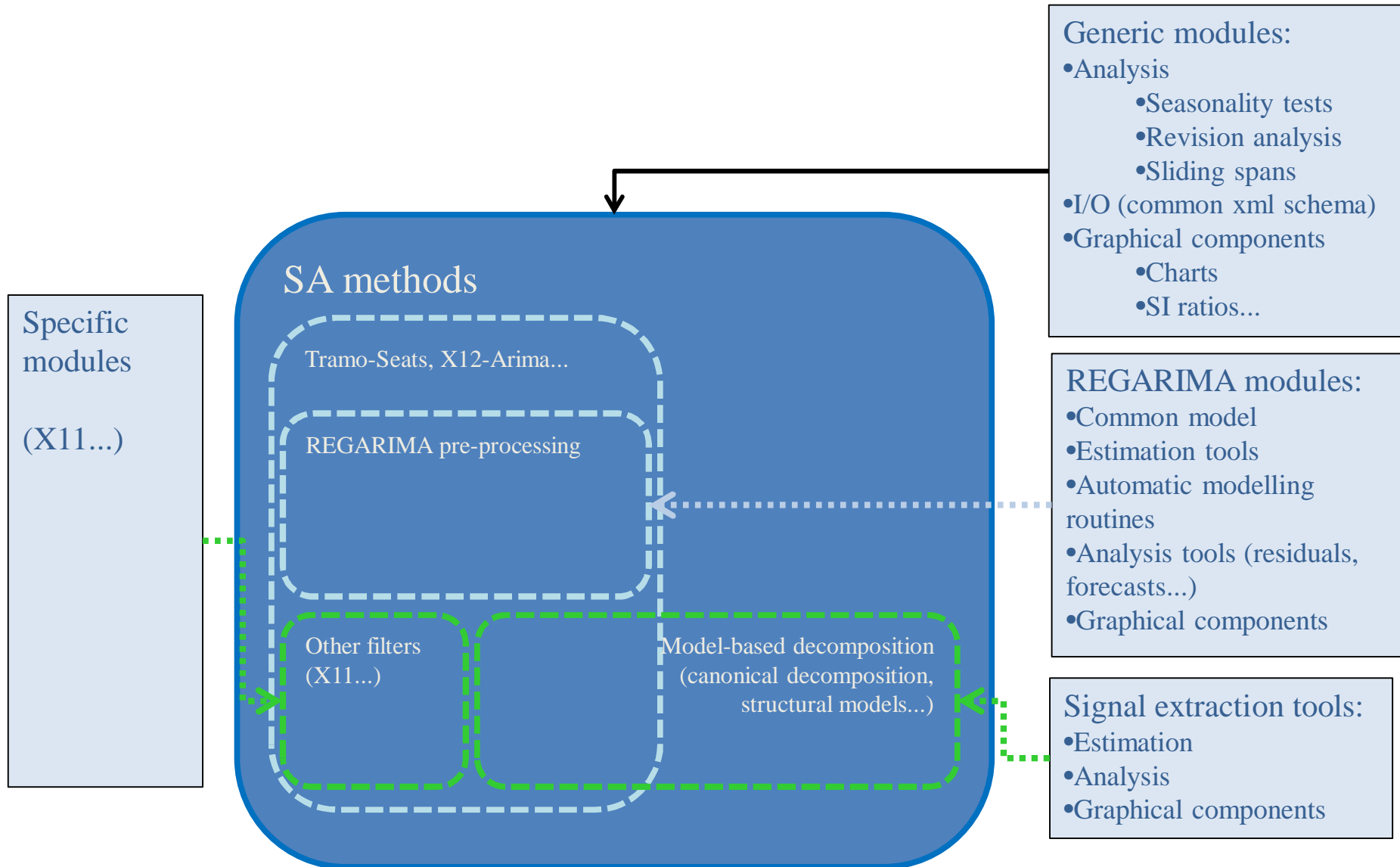
- Tramo-Seats (BDE)
- X12-Arima (USCB)

<https://github.com/jdemetra>

3. Statistical content

Basic data handling	Basic econometrics	Tramo	Benchmarking, temporal disaggregation
Matrix computation	Arima modelling	RegArima	
Complex, polynomials		Structural models...	VAR, Dynamic factor model
Linear filters	Seasonal adjustment	X11	
Function optimization		Seats	High frequency
Basic statistics	Arima, Ucarima	STL	Revisions analysis
Utilities...	State space framework		
Time series, calendars, regression variables...			

4. Seasonal adjustment framework



5. Final remarks

- JD+ is a complete re-factoring of Tramo-Seats and of X12-Arima in an open OO framework. In some cases, the new algorithms may lead to (usually slightly) different results .
- JD+ is also designed for the handling of related time series problems, especially through a rich state space library.
- By developing it as an open source solution, we have tried to create an environment appropriate to external collaborations.