

Workshop

AMiDST
— TOOLBOX

Demo and tutorial

Annual AMIDST meeting
Trondheim, June 28 2016

Introduction

Downloading and setting up material
for the tutorial

System Requirements



- Check your java version:

```
$ java -version
```

- <http://www.oracle.com/technetwork/java/javase/downloads/>



IntelliJ IDEA

- <https://www.jetbrains.com/idea/>

Setting up

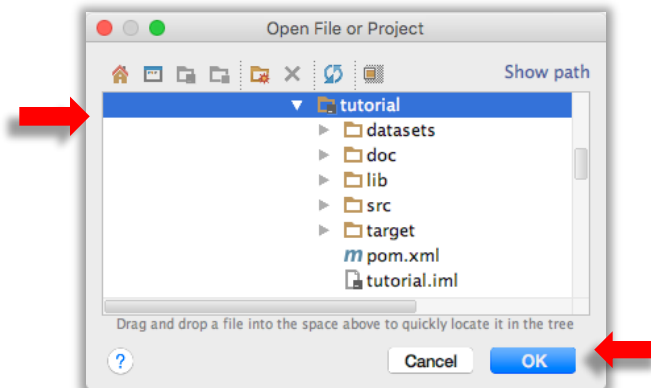


- **Step 1:** Download the example project

```
$ git clone https://github.com/amidst/tutorial.git
```

(or go to <https://github.com/amidst/toolbox>)

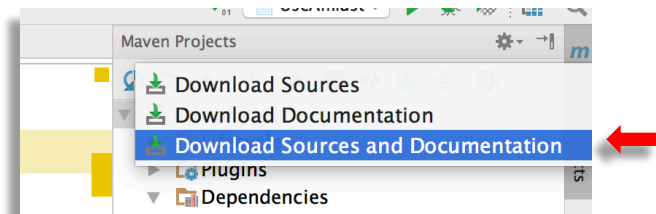
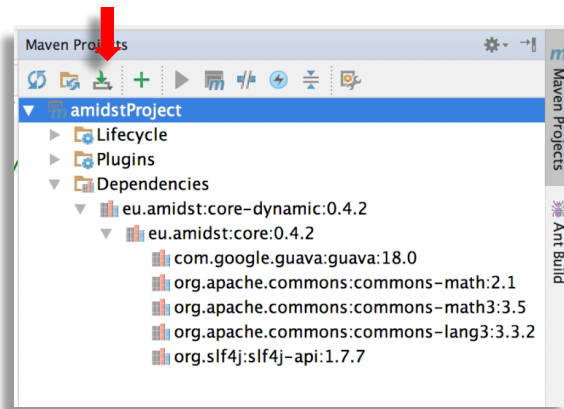
- **Step 2:** Open the downloaded project with IntelliJ:



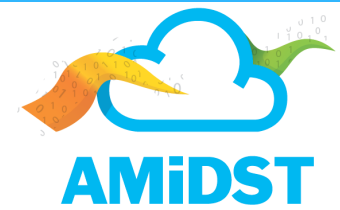
Setting up



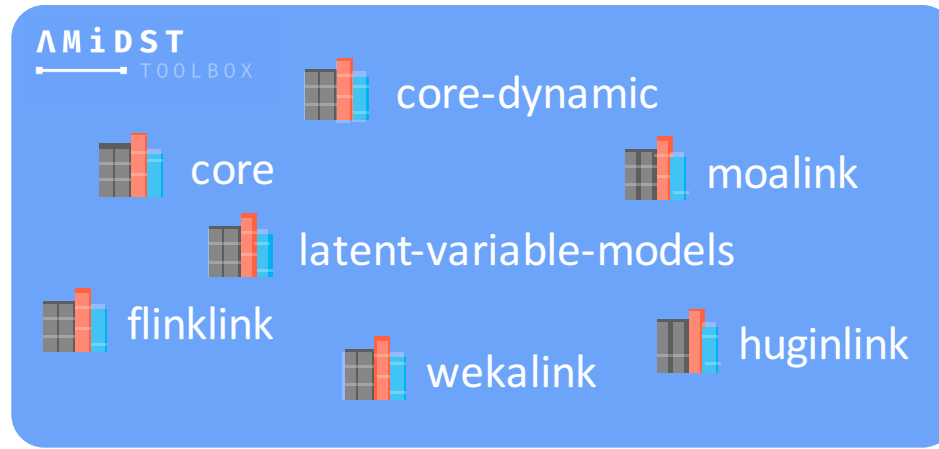
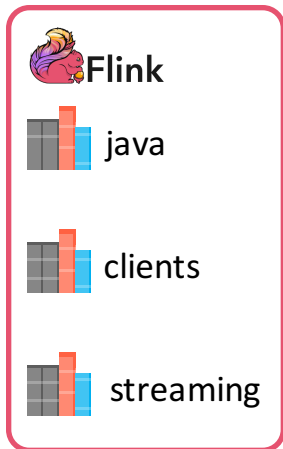
■ Step 3: Download source code and javadoc



Overview



tutorial



Project Structure





- The downloaded project contains:

 **datasets:** ARFF files used in the tutorial

 **doc:** these slides

 **lib:** hugin library

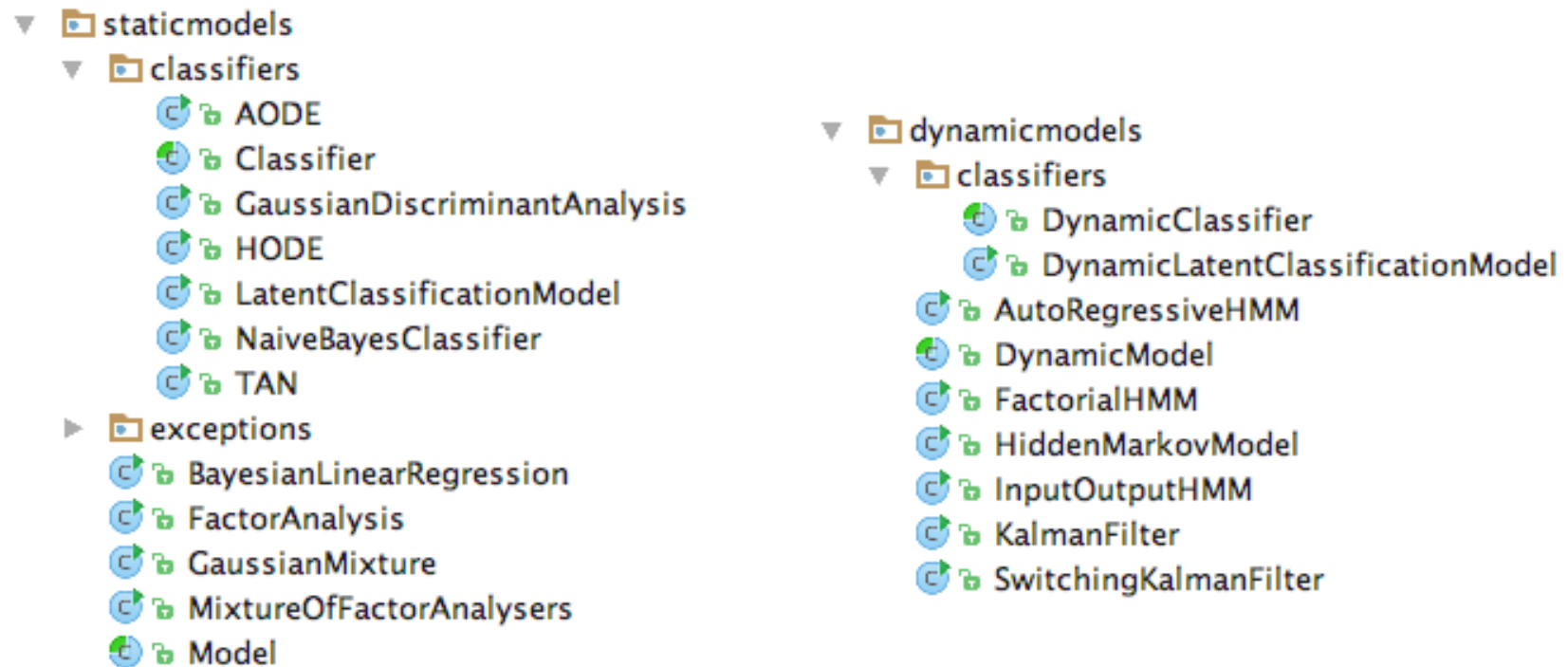
 **src/main/java:** code examples using 

 **pom.xml:** maven dependencies definition

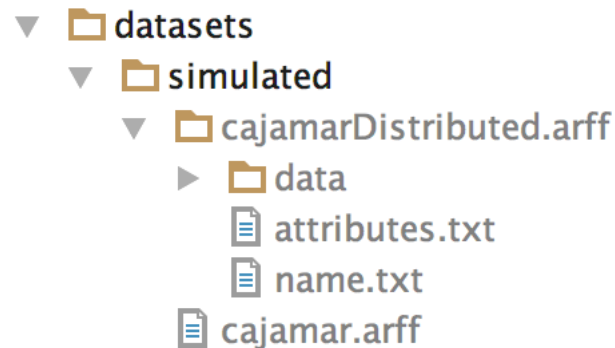
Latent-variable-models



- The module **latent-variable-models** contains a large set of classes for easily learning some of the models in the literature.



- We will see some examples using two datasets:



- The data can be generated with `CreateCajamarDataContinuous.java`

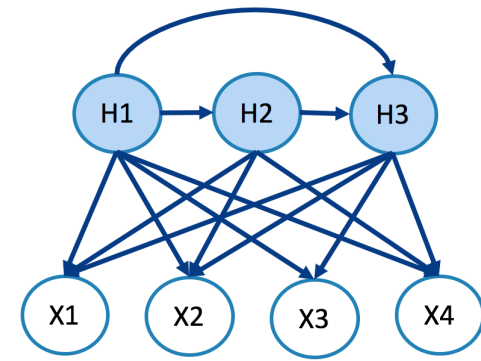
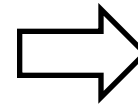
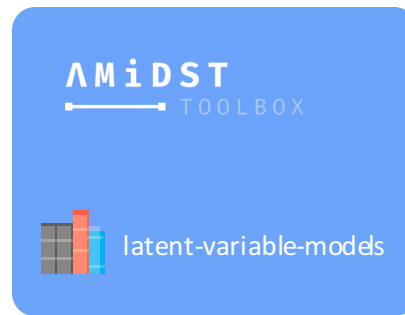
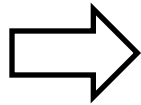
Static Models

A set of code examples for **easily** learning
and making inference with static PGMs

Static models (learning)



ARFF file



Static BN

Static models (learning)



//Load the datastream

```
String filename = "datasets/simulated/cajamar.arff";  
DataStream<DataInstance> data = DataStreamLoader.open(filename);
```

//Learn the model

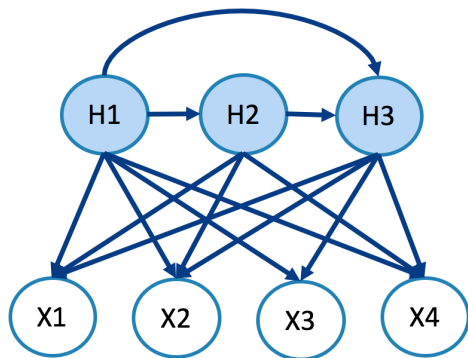
```
Model model = new FactorAnalysis(data.getAttributes());  
model.updateModel(data);  
BayesianNetwork bn = model.getModel();
```

```
System.out.println(bn);
```

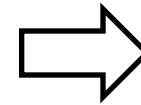
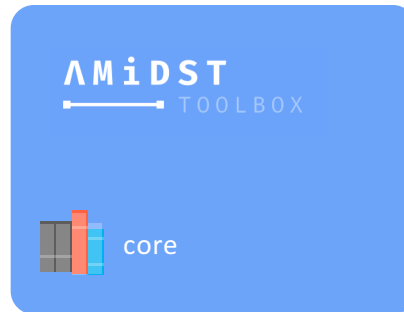
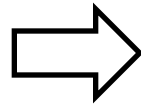


StaticModelLearning.java

Static models (save to disk)



Static BN



.bn file
.net file

Static models (save to disk)



// Save with .bn format

```
BayesianNetworkWriter.save(bn, "networks/simulated/exampleBN.bn");
```

// Save with hugin format

```
BayesianNetworkWriterToHugin.save (bn, "networks/simulated/exampleBN.net");
```

- Note: make sure that you have the following files in your classpath:
 - hgapi83_amidst-64.jar
 - libhgapi83_amidst-64.jnilib
- For adding folders to your class path:

```
-Djava.library.path="..."
```

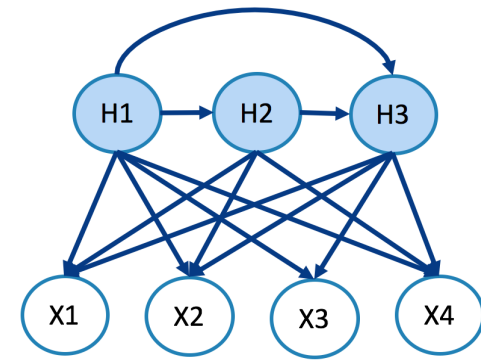
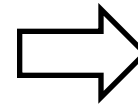
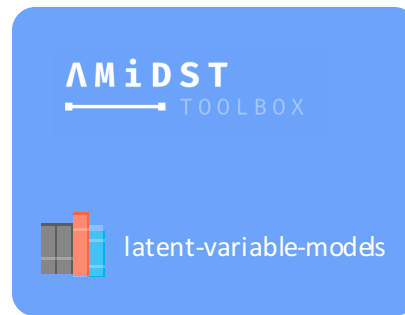
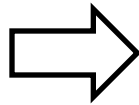


StaticModelLearning.java

Static models (learning from flink)



ARFF folder



Static BN

Static models (learning from flink)



//Load the datastream

```
String filename = "datasets/simulated/cajamarDistributed.arff";
```



```
final ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();  
DataFlink<DataInstance> data = DataFlinkLoader.loadDataFromFolder(env, filename, false);
```

//Learn the model

```
Model model = new FactorAnalysis(data.getAttributes());
```

```
model.updateModel(data);
```

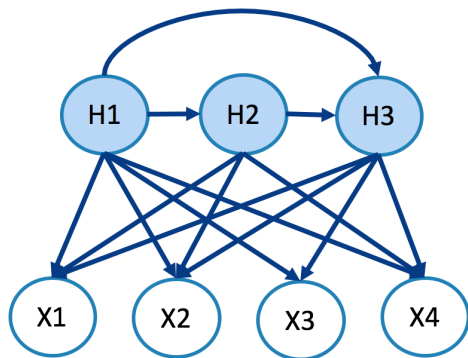
```
BayesianNetwork bn = model.getModel();
```

```
System.out.println(bn);
```

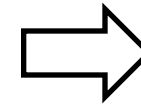
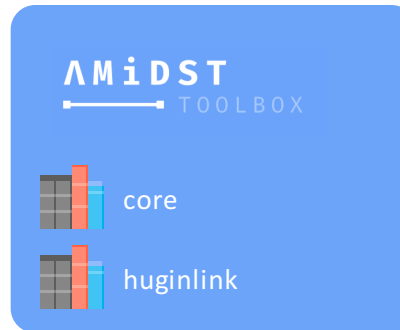
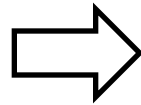


StaticModelFlink.java

Static models (Inference)



Static BN



$P(X2 | H1=0.6)?$

Answer to a
query

Static models (inference)



- Add the following code after learning the model

//Variables of interest

```
Variable varTarget = bn.getVariables().getVariableByName("LatentVar1");  
Variable varObserved = null;
```

//we set the evidence

```
Assignment assignment = new HashMapAssignment(2);  
varObserved = bn.getVariables().getVariableByName("Income");  
assignment.setValue(varObserved, 0.0);
```

//we set the algorithm

```
InferenceAlgorithm infer = new VMP();  
infer.setModel(bn);  
infer.setEvidence(assignement);
```

```
new HuginInference();  
new ImportanceSampling();
```

//query

```
infer.runInference();  
Distribution p = infer.getPosterior(varTarget);  
System.out.println("P(LatentVar1 | Income=0.0) = "+p);
```



StaticModelInference.java

Parallel TAN (Hugin/AMIDST)



- **Hugin:** learn the structure with a subsample of the data
- **AMIDST:** learn the parameters in AMIDST using the whole data.

```
ParallelTAN tan = new ParallelTAN();  
tan.setNumCores(4);  
tan.setNumSamplesOnMemory(1000);  
tan.setNameRoot(var01);  
tan.setNameTarget(classVar);  
BayesianNetwork model = tan.learn(data);
```

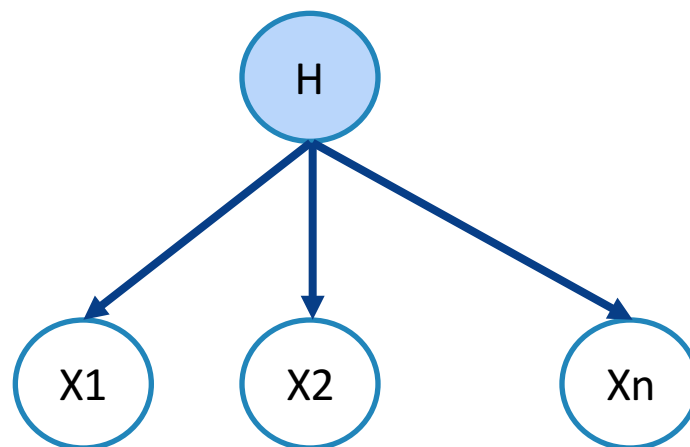


ParallelTANExample.java

Static models (practice)



- Create your custom model: gaussian mixture



Discrete hidden variable

Continuous variables

- Assume that observed variables are not connected

Static models (practice)



- Some tips:

```
public class CustomGaussianMixture extends Model{
```

```
    public CustomGaussianMixture(Attributes attributes) throws WrongConfigurationException {  
        super(attributes);  
        //TODO: Write the constructor code here  
    }
```

```
    @Override
```

```
    protected void buildDAG() {  
        //TODO: Write the code building a DAG for your custom model  
    }  
}
```

- Useful methods:

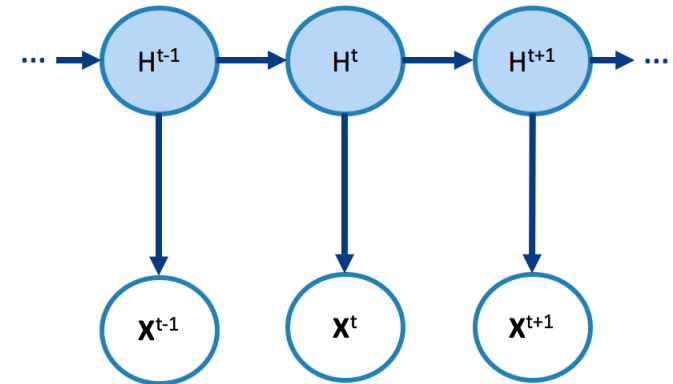
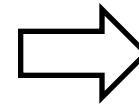
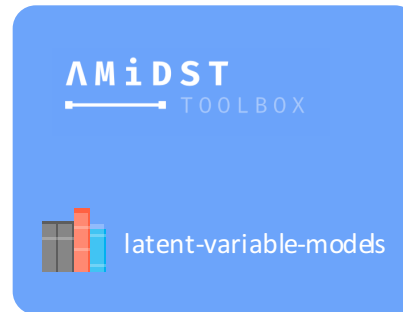
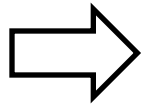
```
public Variable Variables::newMultinomialVariable(String name, int nOfStates)
```

```
public List<ParentSet> DAG::getParentSets()
```

Dynamic Models

A set of code examples for **easily** learning
and making inference with dynamic PGMs

Static models (learning)



ARFF file

Dynamic BN

Dynamic models (learning)



//Load the datastream

String filename = **"datasets/simulated/cajamar.arff"**;

DataStream<DynamicDataInstance> data = DynamicDataStreamLoader.loadFromFile(filename);

//Learn the model

DynamicModel model = **new** HiddenMarkovModel(data.getAttributes());

model.updateModel(data);

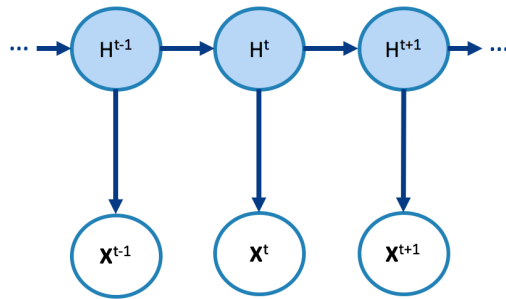
DynamicBayesianNetwork dbn = model.getModel();

System.**out**.println(dbn);

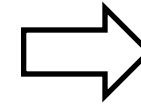
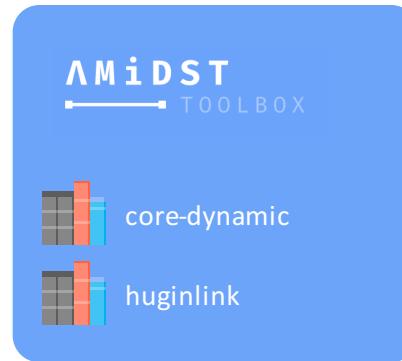
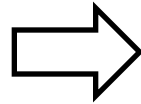


DynamicModelLearning.java

Static models (save to disk)



Dynamic BN



.dbn file
.net file

Dynamic models (save to disk)



// Save with .bn format

```
DynamicBayesianNetworkWriter.save (bn, "networks/simulated/exampleDBN.dbn");
```

// Save with hugin format

```
DynamicBayesianNetworkWriterToHugin.save(bn, "networks/simulated/exampleDBN.net");
```

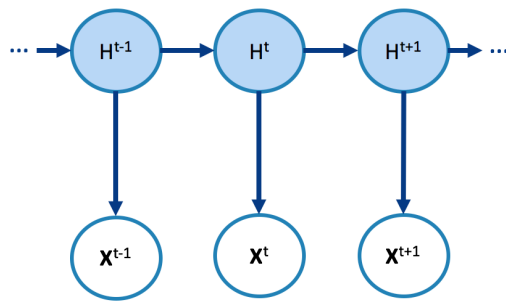
- Note: make sure that you have the following files in your classpath:
 - hgapi83_amidst-64.jar
 - libhgapi83_amidst-64.jnilib
- For adding folders to your class path:

```
-Djava.library.path="..."
```

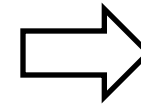
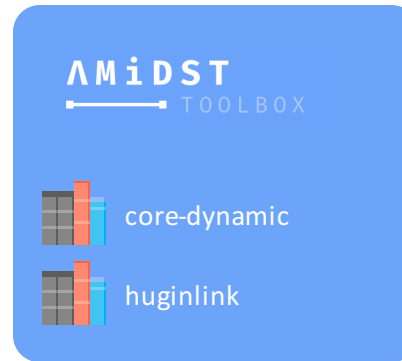
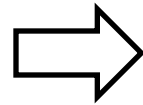


DynamicModelSaveToDisk.java

Static models (save to disk)



Dynamic BN



$$P(H^t | H^{t-1} = h, X^t = x)$$

$$P(H^{t+5} | H^{t-1} = h, X^t = x)$$

query

Dynamic models (inference)



- Add the following code after learning the process

```
//Testing dataset
String filenamePredict = "datasets/simulated/cajamar.arff";
DataStream<DynamicDataInstance> dataPredict = DynamicDataStreamLoader.loadFromFile(filenamePredict);
//Select the inference algorithm
InferenceAlgorithmForDBN infer = new FactoredFrontierForDBN(new VMP());
infer.setModel(dbn);
Variable varTarget = dbn.getDynamicVariables().getVariableByName("discreteHiddenVar");
UnivariateDistribution posterior = null;

//Classify each instance
int t = 0;
for (DynamicDataInstance instance : dataPredict) {

    infer.addDynamicEvidence(instance);
    infer.runInference();
    posterior = infer.getFilteredPosterior(varTarget);
    System.out.println("t="+t+", P(discreteHiddenVar | Evidence) = " + posterior);

}
```



```
new HuginInference();
new ImportanceSampling();
```



DynamicModelInference.java

Dynamic models (inference)



- For predicting 5 steps ahead, replace:

```
posterior = infer.getFilteredPosterior(varTarget);
```



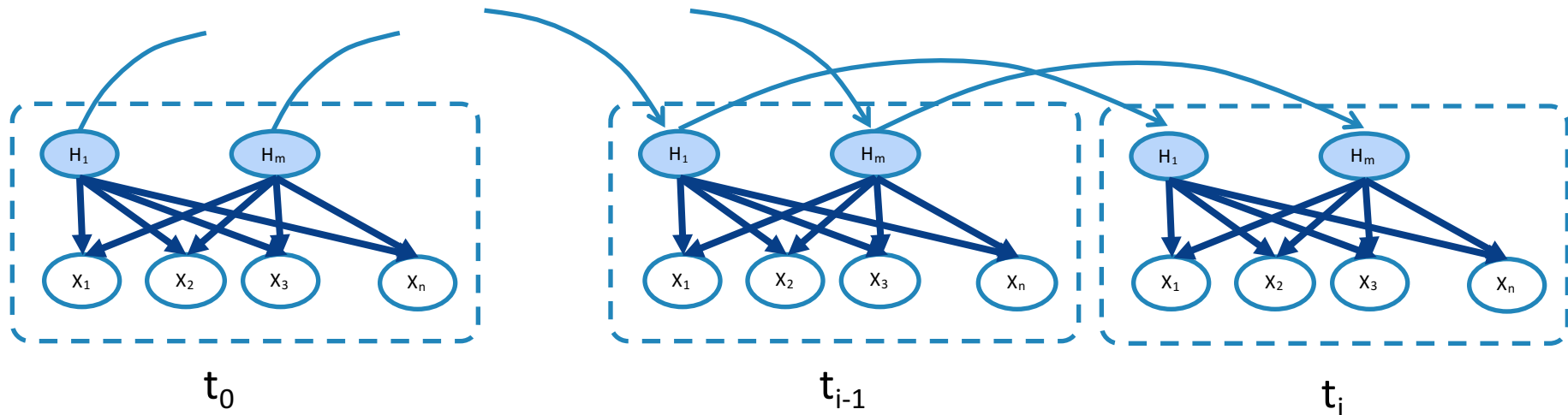
```
posterior = infer.getPredictivePosterior(varTarget, 5);
```



DynamicModelInference.java

Dynamic models (practice)

- Create your custom dynamic model: Kalman filter



- Assume that hidden variables are not connected among them
- All the variables are continuous

Dynamic models (practice)



- Some tips:

```
public class CustomKalmanFilter extends DynamicModel {  
  
    public CustomKalmanFilter(Attributes attributes) throws WrongConfigurationException {  
        super(attributes);  
        //TODO: Write the constructor code here  
    }  
    @Override  
    protected void buildDAG() {  
  
    }  
}
```

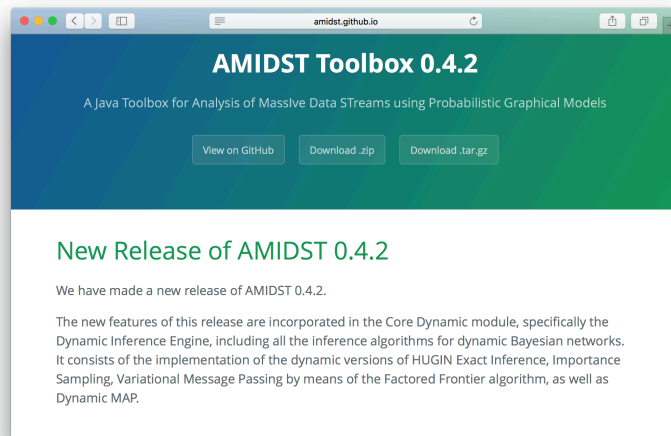
- Useful methods:

```
public DynamicVariable DynamicVariables::newGaussianDynamicVariable (String name)
```

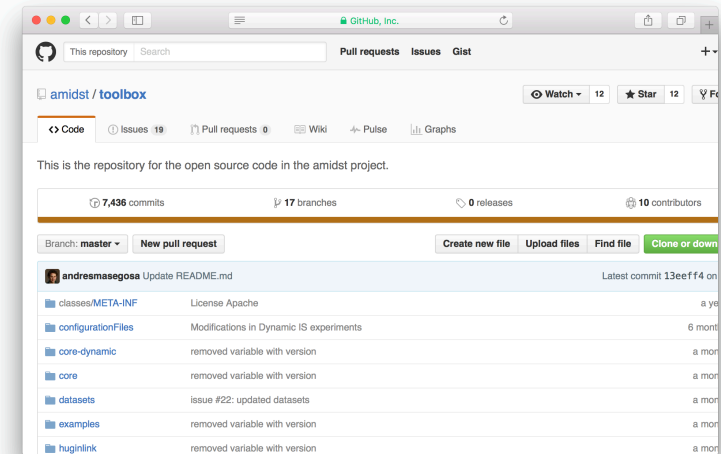
```
public Variable DynamicVariable ::getInterfaceVariable()
```

```
public ParentSet DynamicDAG::getParentSetTimeT(Variable var)
```

Important URLs



<http://amidst.github.io/toolbox/>
(Documentation, tutorials and more)



<https://github.com/amidst/toolbox>
(source code)



Thanks for your attention