Relatório DS1 - OutputAnalyser

Universidade Federal de Santa Catarina - UFSC
INE5425 - Modelagem e Simulação
Enzo Bassani (20200398)
Tulio Scarabelot Bardini (20200424)
Vitor Hugo Homem Marzarotto (20200426)

QuickStart

O Output Analyser desenvolvido pode ser encontrado em "/source/tools/outputAnalyser". Junto do código fonte, pode ser encontrado o arquivo main.cpp, no qual há um script que exemplifica o uso da ferramenta. Para utilizá-lo, execute os seguintes comandos no diretório mencionado:

```
make
./a.out
```

Como utilizar

Esse tópico cobre a utilizacão das principais funcionalidades de ferramenta. Todos os códigos de exemplo exibidos fazem parte do script de exemplo disponibilizado.

Inicialização

O OutputAnalyser é a principal classe da ferramenta desenvolvida, de forma que todas as funcionalidades requisitadas são acessíveis a partir dele. Portanto, a primeira etapa consiste em instanciar e configurar o OutputAnalyser.

```
#include "OutputAnalyser.h"

using namespace std;

int main() {
OutputAnalyser outputAnalyser("Files/timevalue.txt");
outputAnalyser.setReplication(1);
}
...
```

Como pode ser visto acima, basta especificar o arquivo de entrada e a replicação que se deseja analisar.

Síntese Estatística

A classe StatisticalSynthesis é a responsável por realizar a análise sintática dos dados da replicação, como cálculo de média, desvio padrão, mediana, moda, máximo, mínimo, quartil, decil, etc. Seu uso é exemplificado abaixo:

```
...
auto ss = outputAnalyser.statisticalSynthesis();

double min = ss->min();
double max = ss->max();
double avg = ss->average();
auto movingAverage = ss->movingAverage(5);
double var = ss->variance();
...
```

Histograma

A classe Histogram é responsável pelo cálculo e apresentação do histograma. Ela é acessível a partir do StatisticalSynthesis e recebe o número de classes como parâmetro de entrada. Também fornece métodos estáticos para calcular o número de classes pelo método de Sturges ou raiz quadrada.

```
...
int numClasses = Histogram::sturges(outputAnalyser.reader().size());
Histogram histogram = ss->getHistogram(numClasses);
...
```

Fitter

Para os testes de ajustes de curva, utiliza-se a classe Fitter. As distribuições possíveis são: Normal, Lognormal, Weibull, Erlang, Exponencial e Uniforme. Para realizar o teste, basta passar o nível de confiança e os parâmetros necessários para a geração da distribuição, como é mostrado abaixo:

```
...
Fitter* fitter = outputAnalyser.fitter();

bool isNormal = fitter->isNormalDistributed(0.05, 4);
bool isLognormal = fitter->isLognormalDistributed(0.05, 4);
bool isWeibull = fitter->isWeibullDistributed(0.05, 4, 1, 1);
bool isErlang = fitter->isErlangDistributed(0.05, 2, 1);
```

```
bool isExpo = fitter->isExpoDistributed(0.05, 1);
bool isUniform = fitter->isUniformDistributed(0.05);
...
```

Correlograma

Por fim, a classe Correlogram é responsável pelo cálculo do vetor de autocorrelação. Ela é acessível a partir do outputAnalyser:

```
...
auto correlogram = outputAnalyser.getCorrelogram(0, -1);
...
```

Testes

Os testes estão disponíveis na pasta UnitTests, no arquivo runTests.cpp. Para executá-los, basta utilizar o comando “make test”.

Notas

- ## Reader:

Todas as classes desenvolvidas acessam os dados da replicacão através de uma interface Reader_if. Duas classes implementam essa interface: ReaderImpl e SortedReader. A primeira realiza a leitura dos dados a partir do arquivo, sendo a classe comumente utilizada pelo OutputAnalyser para ler os dados, escolher o arquivo e a replicação desejada. A segunda é inicializada a partir de um vetor e salva uma cópia ordenada dos dados.

A arquitetura do OutputAnalyser é tal que todas as suas dependências acessam os mesmos readers, não havendo necessidade de copiá-los. Ao atualizar o arquivo ou a replicação sendo analisada, todas as subclasses também passam a ler daquele arquivo (uma vez que o reader, comum a todas, foi atualizado).

- ## Fitter:

Como já mencionado, o Fitter implementa os testes de ajuste de curva na amostra de referência. Contudo, a classe tem dois tipos de testes, o isDistributed e o fitDistributed. No isDistributed os testes de ajuste são realizados com base nas médias e desvio padrão da amostra, referente ao nível de confiança definido.

Quando estamos usando métodos do tipo fitDistributed, passamos os parâmetros por meio de ponteiros, juntamente com o parâmetro de erro quadrático. Essas funções não possuem retorno, pois o acesso de seu resultado se dará pelo próprio ponteiro do erro quadrático.

- ## Correlograma:

O Correlograma gera um vetor de autocorrelação dado uma amostra de uma variável sobre o tempo. A autocorrelação é gerada a partir de um intervalo de tempo. Por convenção, o tempo final do intervalo pode ser expresso por -1, indicando que é desejado usar até o último tempo de coleta da amostra.