1º trabalho de Física Computacional (2021-22, P3)

Entrega do trabalho

- Até às 20H de dia 26 de Março de 2022 (sábado) através do svn, **única e exclusivamente**.
- Não se esqueçam de fazer `commit` de todos ficheiros com excepção dos ficheiros `*.o` `*.exe`
- A operação `svn status` permite identificar os ficheiros ainda não `commited` ou que não estejam ainda sob controlo de svn.
- Caso existam entregas posteriores às 20H05 (atrasos superiores a 5 minutos de tolerância), a regra de desconto em valores [0,20] que será aplicada à nota do trabalho será a seguinte:

  $D = T * 0.2$
  D: desconto em valores [0,20]
  T: tempo de atraso de entrega em minutos, a partir das 20H05 (fim da tolerância)

Organização das pastas de trabalho

Em cada grupo, foi criada a pasta **trab01** que contém as seguintes pastas e ficheiros (não se esqueça de começar por fazer `svn update`):

```
trab01/src ..................... [user classes]
      /main .................... [main programs]
      /bin ..................... [object files, executables]
      Folha_Respostas.pdf ..... [folha de respostas]
      T1.pdf ................... [enunciado do trabalho]
      tDFT_sinal.dat ........... [dados para as perguntas 1. e 2.]
      tDFT_sinalruido.dat ..... [dados para a pergunta 3.]
```

De notar:

- Os programas principais que realizarem devem estar na pasta `main/`
- As classes (ficheiros “header” e “source”) desenvolvidos pelo grupo e necessários à resolução deste trabalho, devem estar na pasta `src/`

Correcção e Cotação

A resolução do trabalho implicará a entrega:

- dos códigos C++ realizados (classes e programa principal)

- da folha de respostas (ficheiro `Folha_Respostas.pdf`) preenchida onde constará a identificação dos elementos do grupo que participaram na realização do trabalho > o software PDFexpert (macOS) e Okular (linux) permitem a escrita no ficheiro pdf

A folha de respostas pode ser preenchida à mão ou no computador e deve ser submetida no svn. Caso seja preenchida à mão proceda à sua digitalização e junte ao svn.

Passos que serão seguidos na correcção do exercício:

- estrutura do código (5%)
- clareza e comentários do código C++ (15%)
- execução do código, resultados e respostas (80%)

Enunciado

Análise de um sinal variável em tempo

Este trabalho tem como base o trabalho 1 proposto para casa (*homework*), assumindo-se assim que a introdução lá realizada deve ser lida.

O ficheiro de dados existente na pasta do grupo com o nome `tDFT_sinal.dat` que irá tratar, possui uma série temporal cuja amostragem foi feita a intervalos regulares de tempo, com $\Delta t = 0.001$ segundos e para um tempo total de 5 segundos.

Deve estruturar o código C++ que vai realizar nas **pastas de trabalho** `trab01/src/` e `trab01/main/`. Assim:

- as declarações das funções que tiver que realizar devem estar no ficheiro `trab01/src/T1func.h`
- o código onde implementa as funções deve estar no ficheiro `trab01/src/T1func.cpp`.
- os programas principais devem estar pasta do grupo `trab01/main/`

[14 valores]

1. Realize um programa em C++, cujo nome deve ser `T1a.cpp` e que deve ser adicionado à pasta `trab01/main/`, que faça a leitura dos dados registados no ficheiro `tDFT_sinal.dat` existente na pasta do trabalho.

   O programa deve seguir o esqueleto abaixo, respeitar o nome das funções e o tipo de argumentos que são passados e realizar as seguintes tarefas:

   - Fazer a leitura do ficheiro de dados, colocando o seu conteúdo em dois arrays: array de valores tempo (t), e array de valores amplitude (A)
   - Fazer o cálculo da média deslizante do sinal usando uma janela de 21 pontos e com o passo de 1 ponto
   - Obter os coeficientes de Fourier
   - Obter o periodograma (densidade de potência espectral)
   - Obter o coeficiente de auto-correlação ρ,

$$\rho_k = \left(\frac{N}{N-k}\right) \frac{\sum_{t=k+1}^{T} \left(A_t - \bar{A}\right)\left(A_{t-k} - \bar{A}\right)}{\sum_{t=1}^{T} \left(A_t - \bar{A}\right)^2}$$

   N : número total de pontos de amostra
   k : desfasamento temporal ($t_0 = k\, t_s$)
   t_s : intervalo de tempo entre medidas do sinal

- Desenhe os seguintes plots:

  a. **auto-correlação .vs. tempo**
  Fazer um gráfico (`TGraph`) auto-correlação .vs. tempo, no intervalo de tempo [0, 0.2] segundos, ligando os pontos com linhas a verde (`kGreen+2`): opção `Draw("AL")`

  > guardar o gráfico num ficheiro de nome `T1a_autocorrelacao.png`

  b. **sinal .vs. tempo , média dos sinais .vs. tempo**
  Fazer um gráfico (`TGraph`) amplitude do sinal .vs. tempo, no intervalo de tempo [0, 0.2] segundos, com o sinal representado por pontos representados por símbolos `MarkerStyle=25` a côr azul (`kBlue+2`), ligados por linhas da mesma côr e a média deslizante dos sinais representada por pontos de côr vermelha (`kRed+2`), com símbolos `MarkerStyle=24` e ligados por linhas da mesma côr.

  > guardar o gráfico num ficheiro de nome `T1a_sinal.png`

  c. periodograma: **densidade espectral .vs. frequência**

  > guardar o gráfico num ficheiro de nome `T1a_periodograma.png`

Leia com detalhe os comentários existentes no programa, complete o código e desenvolva as funções necessárias.

T1a.cpp

```
// function declarations
(...)

// main program
int main() {

  // Read data file to arrays
  const int N = (...);
  double t[N], A[N];
  ReadFile("tDFT_sinal.dat", t, A);

  // Get auto-correlation
  // - function returns vector of pair's with
  //   (time shift, auto-correlation coeff)

  vector<pair<double,double>> vA = GetAutoCorrelation(N, t, A);

  // print 20 1st values of time and auto-correlation
  (...)

  // Get Fourier Coefficients
  // - vC: vector containing pair's
```

```
  //   (frequency(Hz), Fourier coefficients X(fk) )

  vector< pair<double,complex<double>> > vC;
  GetFourierCoeffs(N, t, A, vC);

  // print 6 most important Fourier coefficients using
  // std::for_each

  std::for_each(...);

  // compute periodgram
  // - store periodgram into a vector of pairs
  //   (frequency(Hz), spectral power(fk) )

  vector<pair<double,double>> vP;
  (...)

  ////////////////////////////////////////// drawing with
     ROOT

  TApplication A("A",0,0);
  TCanvas C("canvas", "", 1200,800);

  (...)

}
```

[4 valores]

2. O esboço de programa que se segue deve ser completado, respeitando o nome das funções, como são chamadas e o tipo de variáveis. O programa cujo nome a atribuir deverá ser `trab01/main/T1b.cpp`, pretende reconstruir o sinal contínuo em tempo, com base nas componentes harmónicas detectadas a partir dos resultados obtidos na alínea 1. Os resultados da alínea 1 forneceram o espectro das frequências que compõem o sinal, $P(f_k)$. Claramente é possível ver que existem frequências no espectro com mais importância que outras.

   A reconstrução do sinal em tempo, $s(t)$, a partir do espectro de frequências pode fazer-se tomando a componente *Real* da seguinte expressão (Transformada Inversa de Fourier):

   $$s(t) = \Re\left\{\frac{1}{M}\sum_{i=0}^{M-1} X(f_i)\, e^{j2\pi f_i t}\right\}$$

   em que:

   - M: número de frequências usadas na reconstrução do sinal

   Analisando o espectro de frequências obtido anteriormente pode definir o seu sinal reconstruído somente com o conjunto de frequências dominantes.

   Leia com atenção os comentários no programa abaixo. Complete o programa e desenvolva as funções necessárias que devem ser colocadas tal como dito anteriormente,

na pasta `trab01/src/`: os headers no ficheiro `T1func.h` e o código das funções no ficheiro `T1func.cpp`.

Obtenha o seguinte plot:

- a função do sinal reconstruído em função do tempo para o intervalo [0, 0.2] segundos, com côr `kRed+2`. Sobreponha no mesmo gráfico os pontos da amplitude do sinal original medido, com côr `kBlue+2` e `MarkerStyle=25`.

> Guardar o gráfico num ficheiro com o nome `T1b_sinalrec.png`

T1b.cpp

```
// function declarations
(...)

// main program
int main() {

  // Read data file to arrays

  const int N = (...);
  double t[N], A[N];
  string fname = "tDFT_sinal.dat"; // file name
  ReadFile(fname, t, A);

  // Get Fourier Coefficients
  // - vC: vector containing pair's
  //   (frequency, Fourier coefficients X(fk) )

  vector< pair<double,complex<double>> > vC;
  GetFourierCoeffs(N, t, A, vC);

  // Reconstruct signal in time, f(t), from detected harmonics
  // using a lambda function and a TF1

  auto frec = [(...)](double* x, double* par){
     // x[0]: time (sec)
     // no parameters
     (...)
  };
  auto F = new TF1(...);

  // compare data file amplitudes with reconstructed ones

  cout << "|    time    |    amplitude    |  signal rec    |" <<
     endl;
  for (int i=0; i<20; ++i) {
     cout << t[i] << " " << A[i] << " " << frec(&t[i],nullptr)
        << endl;
  }
```

```
  ////////////////////////////// drawing

  (...)

}
```

[2 valores]

3. Pretende-se agora definir uma estratégia que permita determinar as frequências que compõem o sinal, existente no ficheiro `tDFT_sinalruido.dat` que possui um ruído muito importante. O sinal foi medido com um intervalo de tempo de 0.0025 segundos e um número total de 2000 pontos.
   Faça a visualização do sinal e defina uma estratégia que lhe permita identificar as frequências das harmónicas principais que o compõem.

   Todo o trabalho que decida desenvolver nesta alínea, deve ser colocado no programa `trab01/main/T1c.cpp`.

   Explique detalhadamente o seu raciocínio e os resultados que obteve, na folha de respostas. Os resultados obtidos devem ser verificáveis correndo o programa que realizou.