Manual

O programa é composto por 5 algoritmos: **final.py**, **find_probe.py**, **make_arq.py, find_peak** e **compare_phase.py**.

O algoritmo **final.py** tem como objetivo concentrar os outros quatro algorítmicos, chamando-os na ordem de execução do programa.

O algoritmo **find_probe.py** tem como objetivo converter a nomenclatura dos transcritos na lista para a nomenclatura encontrada em **C1_ritm.txt** e **C2_ritm.txt**, arquivos que contém os dados de expressão dos transcritos nas coletas de inverno (CI) e verão (CV), respectivamente. Na primeira parte do programa os nomes dos transcritos a serem convertidos são inseridos em uma lista. Os nomes dos transcritos estão no arquivo **lista.txt**. Na segunda parte do programa, cada linha do arquivo que contém a conversão da nomenclatura (**C1_prob.txt**) é comparado com a lista de transcritos. Quando o nome do transcrito é encontrado no arquivo de conversão, a nomenclatura convertida é adicionada em um novo arquivo, chamado **prob_list_C1.txt**.

O algoritmo **make_arq.py** tem dois objetivos: (1) copiar os dados de expressão em **C1_ritm.txt** e **C2_ritm.txt** dos transcritos desejados escritos em **prob_list_C1.txt** para um novo arquivo**;** e (2) calcular o zero relativo para os transcritos desejados em cada uma das coletas. O zero relativo se define como a média entre a maior intensidade e a menor intensidade entre todos os transcritos desejados em uma coleta. Primeiramente o algoritmo executa os dados relacionados à coleta de inverno e depois executa os dados relacionados à coleta de verão, porém a execução do algoritmo para os dois grupos de dados é igual. Inicialmente, o algoritmo transforma o arquivo **prob_list_C1.txt** com os

nomes dos transcritos de interesse em uma lista. Em seguida, o algoritmo percorre as linhas do arquivo **C1_ritm.txt.** Caso a linha do arquivo **C1_ritm.txt** corresponda a um dos transcritos na lista de transcritos de interesse, a linha de **C1_ritm.txt** é escrita em **expressao_C1.txt**. Ao mesmo tempo, caso na linha do arquivo **C1_ritm.txt** tenha uma intensidade maior ou menor que as armazenadas nas variáveis **maior_C1** e **menor_C2**, esses valores substituem os contidos nas variáveis. Após percorrer todo o arquivo **C1_ritm.txt**, é escrito no arquivo **zero_C1.txt** o valor do zero relativo para os transcritos de interesse na coleta de inverno. O mesmo é repetido para a coleta de verão.

O algoritmo **find_peak.py** tem como objetivo identificar os picos dos transcritos de interesse. O algoritmo é primeiro chamado em **final.py** com os dados de expressão da coleta de inverno e depois chamado com os dados de expressão da coleta de verão. Inicialmente, o algoritmo armazena o valor do zero relativo calculado em **make_arq.py** na variável **zero**. Em seguida, o algoritmo pega cada linha do arquivo **expressao_C1.txt** (na coleta de inverno) e **expressao_C2.txt** (na coleta de verão) para identificar o pico. Dois tipos de pico podem ser identificados:

1) aqueles que têm um pico fragmentado, com parte no início da coleta e outra no fim da coleta, identificados pela condição:

```
if linha1[inicio] > zero and linha1[fim] > zero
```

Em que **inicio** é a variável que contém a localização da primeira intensidade de expressão, correspondente ao primeiro horário de coleta (1,5 h antes do amanhecer), e **fim** é a variável que contém o último dado de expressão,

correspondente ao último horário de coleta (20 h após o amanhecer). Caso o transcrito tenha uma expressão acima de **zero** tanto no primeiro quanto no último horário de coleta, é considerado um pico com início no último horário de coleta e fim no primeiro horário de coleta, uma vez que a expressão é cíclica.

2) aqueles com pico contínuo durante a coleta, identificados pela exceção da condição em 1.

Após identificado o tipo de pico, o algoritmo identifica quão longo é o pico, a intensidade e o horário do ápice do pico e o horário de início e fim do pico, as variáveis **pico1**, **maior1**, **zt_maior1**, **x1** e **x2**, respectivamente. Todas essas informações são armazenadas no vetor **resultado1**. O algoritmo continua a verificar as intensidades, identificando se há mais de um pico na expressão e suas características. Caso a intensidade não corresponda a um pico pela condição:

```
if linha1[zt] > zero,
```

a intensidade corresponde a um vale. Assim como para o pico, é identificado no vale o quão longo ele é, sua intensidade e horário de mínimo, e o horário de início e fim do vale, as variáveis **vale2**, **menor2**, **zt_menor2**, **y1** e **y2**, respectivamente. Os dados do vale são armazenados no vetor **resultado2**. São somente considerados picos ou vales quando a expressão do pico ou do vale ocorre em dois ou mais horários de coletas, de forma a evitar que ruídos sejam considerados picos ou vales. Além das variáveis já mencionadas, são importantes também as variáveis **n_pico1** e **n_vale2**, que quantificam o número de picos e vales na expressão do transcrito, respectivamente. O vetor **perfil1**

traduz a expressão do transcrito em 0s e 1s, em que o 0 corresponde a um horário de vale e 1 corresponde a um horário de pico. Por fim, o algoritmo armazena os dados de **resultado1**, **resultado2**, **n_pico1**, **n_vale2** e **perfil1** nos arquivos **dados_C1.txt** para a coleta de inverno e **dados_C2.txt** para a coleta de verão. Essa análise se repete para todas as linhas do arquivo com os dados de expressão.

O algoritmo **compare_phase.py** seleciona os transcritos que têm somente um pico e são expressos em CI e CV e os compara. As principais características comparadas são a duração do pico, o horário do valor máximo, e a amplitude de expressão. Também é identificada e comparada a presença de um pico ou um vale em ZT18 em CI. O arquivo final do algoritmo (**arq_final.txt**), contém como informações:

1 - **O nome do transcrito.** Transcritos com mais de um pico identificado possuem somente seus nomes na linha (1ª coluna);

2 - **O ZT de início e fim do pico para CI e CV**, com os horários normalizados (2ª a 5ª coluna);

3 - **A duração do pico**, que foi identificada subtraindo o valor do ZT de fim pelo ZT de início do pico. Caso o pico inicie e termine no mesmo dia (identificado pela condição `if y2 > y1` para CI ou `if b2 > b1` para CV), o valor do período foi calculado por ZT fim – ZT início. Caso o pico inicie em um dia e termine no seguinte, o valor do período foi calculado subtraindo 24 h pelo ZT de início e somando o valor do ZT de fim. Os dados estão na 6ª e 7ª coluna;

4 - **Diferença da duração do pico**, calculado subtraindo os valores de duração do pico de CI e CV, sendo mostrado o valor absoluto (8ª coluna);

5 – **Fase**, o ZT normalizado em que se encontra o ápice de expressão do transcrito (o ZT na qual o nível de transcrição é máximo) para CI e CV, que caracteriza o horário do ponto máximo do transcrito (9ª e 10ª coluna);

6 - **Diferença de fase entre CI e CV**, calculado pela subtração entre a fase de CI menos a fase de CV. Caso o valor da diferença seja negativo, o transcrito é expresso mais cedo em CI. Caso o valor da diferença seja positivo, o transcrito é expresso mais cedo em CV. Caso o valor absoluto da diferença seja maior que 12 h, ajustou-se o valor da diferença de forma que correspondesse a um período dentro do intervalo [-12,12], com a fase de CI como referência. Por exemplo, se a fase de CI for ZT16 (~22 h em um amanhecer 6 h) e a fase de CV for ZT1 (~7 h em um amanhecer 6h), a diferença entre a fase será 15 h. Isso significaria que a fase pico do transcrito em CI é 22 h e retrocede 15 h para ser expresso 7 h em CV. Ao ajustar para um intervalo de [-12,12], o sentido da diferença de fase muda. O transcrito de CI é expresso e tem fase 22 h, no entanto, a fase avança 9 h em CV, sendo expressa 7 h. No algoritmo, esse ajuste é feito por condições. Se a diferença de fase for maior que 12h, temos um caso em que CV avança em relação a CI, como no exemplo dado. O ajuste é feito pela equação: `fase = -24 + fase`. Se a diferença de fase for menor que -12, ocorre o oposto, CV retrocede em relação a CI. O ajuste é feito pela equação: `fase = 24 + fase`. Os dados se encontram na 11ª coluna.

7 - **Os ápices de expressão não normalizados**, identificados pelos algoritmos que determinam os transcritos rítmicos (12ª e 13ª coluna).

8 - **A diferença de fase pelos ápices de expressão** identificados pelos algoritmos que determinam os transcritos rítmicos. Calculado subtraindo o ápice de CI pelo ápice de CV. Os dados se encontram na 14ª coluna.

9 - **Presença de pico, vale ou nenhum dos anteriores no ZT 18**. Foi considerado um pico extra em ZT18 quando os horários de coleta antes e depois do ZT18 possuem intensidades abaixo do zero relativo e no ZT18 intensidade acima do zero relativo. Foi considerado um vale extra em ZT18 quando os horários de coleta antes e depois do ZT18 possuem intensidades acima do zero relativo e no ZT18 intensidade abaixo do zero relativo. A identificação dos picos e vales no ZT18 se deram pelo perfil de expressão determinado em **find_peak.py** com 0s e 1s, em que 0s representam intensidade abaixo do zero relativo e 1s representam intensidades acima do zero relativo. Dessa forma, picos em ZT18 se caracterizam pelo padrão 010 nas posições 9,10 e 11, respectivamente, na variável **perfil**. Vales em ZT18 se caracterizam pelo padrão 101 nas posições 9,10 e 11, respectivamente, na variável **perfil**. Nas colunas 15 e 17 do **arq_final.txt**, a informação se apresenta pelas *strings* ‘pico’, ‘vale’ e ‘nada’. Nas colunas 16 e 18, essa informação é traduzida em números inteiros, em que o pico é representado por 1, o vale por -1 e a ausência de pico e vale por 0.

10 - **Amplitude de CI e CV**, calculada pela subtração da maior intensidade de expressão pela menor intensidade de expressão. Como o mínimo de expressão tende a ser negativo, a amplitude se traduz na soma da intensidade máxima pela mínima da expressão do transcrito. Para a determinação do mínimo de expressão, considerou-se possível até “três” vales aparentes. Um vale seria

formado caso a expressão do transcrito seja fragmentada, com parte no início e outra no fim da coleta. “Dois vales” seriam possíveis caso a expressão seja contínua durante a coleta, formando um vale no início e outro no fim da coleta. “Três vales” seriam possíveis caso haja um pico de expressão extra em ZT18, quando é o pico se encontra no meio da coleta. Desta forma, um vale seria no início da coleta, outro no fim da coleta antes do ZT18 e outro após ZT18. Dentre esses possíveis vales, foi identificado o que possui o menor valor de intensidade. Os dados se encontram nas 19ª e 20ª colunas.

11 - **Razão da amplitude**, calculada pela razão da amplitude de CI pela amplitude de CV. Valores menores que 1 indicam que a amplitude de CV é maior que a de CI (21ª coluna).

12 - **Taxa de transcritos de acordo com a fase em CI e CV**. Os horários de coleta foram numerados de 0 a 11 e para cada um deles foi calculado quantos transcritos têm fase nesse horário. A quantidade de transcritos por horário de coleta foi dividida pelo total de transcritos analisados no algoritmo. Os dados se encontram abaixo dos dados descritos nos tópicos 1 a 11, intitulados “Porcentagem de picos em C1” e “Porcentagem de picos em C2”.

13 - **Taxa de transcritos de acordo com a fase em CI e CV quando eles têm pico em ZT18.** Para cada horário de coleta, foi calculado a taxa de quantos transcritos têm pico extra em ZT18 e ápice de expressão nessa fase. Os dados se encontram abaixo dos dados descritos no tópico 12, intitulados “Porcentagem de picos em C1 quando ZT18 tem pico” e “Porcentagem de picos em C2 quando ZT18 tem pico”.

14 - **Taxa de transcritos de acordo com a fase em CI e CV quando eles têm vale em ZT18.** Para cada horário de coleta, foi calculado a taxa de quantos transcritos têm vale extra em ZT18 e ápice de expressão nessa fase. Os dados se encontram abaixo dos dados descritos no tópico 12, intitulados “Porcentagem de picos em C1 quando ZT18 tem vale” e “Porcentagem de picos em C2 quando ZT18 tem vale”.

15 - **Número de transcritos com pico, vale e ausência de pico e vale em ZT18.** Os dados se encontram abaixo dos dados descritos no tópico 14.

O programa é composto por 13 arquivos em formato txt, que incluem o arquivo inicial e final, arquivos de transição que transferem a informação entre os algoritmos, arquivos com dados necessários para o funcionamento do algoritmo e o arquivo com instruções do programa.

O arquivo **lista.txt** é o arquivo de entrada, em que o usuário insere a lista de transcritos a serem analisados. Os transcritos inseridos devem ser expressos tanto em CI quanto em CV. Este arquivo é usado no algoritmo **find_probe.py**.

O arquivo **C1_probe.txt** contém as informações para a conversão da nomenclatura dos transcritos e é usado no algoritmo **find_probe.py**. Os dados no arquivo **lista.txt** devem ter a nomenclatura da primeira coluna de **C1_probe.txt** e são convertidos para a nomenclatura da segunda coluna. Os transcritos listados em **C1_probe.txt** são todos os transcritos expressos em CI.

O arquivo **prob_list_C1.txt** contém a lista de transcritos em **lista.txt** com a nomenclatura convertida por **find_probe.py**. Os dados em **prob_list_C1.txt** são escritos em **find_probe.py** e usados em **make_arq.py.**

Os arquivos **C1_ritm.txt** e **C2_ritm.txt** contém os dados de expressão de todos os transcritos rítmicos em CI e CV, respectivamente. Os horários de coleta estão normalizados. Os arquivos são utilizados em **make_arq.py**.

Os arquivos **expressao_C1.txt** e **expressao_C2.txt** contém os dados de expressão dos transcritos de interesse inseridos em **lista.txt**. O arquivo **expressao_C1.tx**t tem os dados de expressão de CI e **expressao_C2.txt** tem os dados de expressão de CV. Esses dois arquivos são escritos em **make_arq.py** e usados em **find_peak.py**.

Os arquivos **zero_C1.txt** e **zero_C2.txt** contém o valor do zero relativo para CI e CV, respectivamente, para os transcritos de interesse. Além disso, a maior intensidade e a menor intensidade dentre as intensidades dos transcritos de interesse também estão escritas. A média dessas intensidades é usada para calcular o zero relativo. Esses dois arquivos são escritos em **make_arq.py** e usados em **find_peak.py**.

Os arquivos **dados_C1.txt** e **dados_C2.txt** as informações sobre os picos e vales para cada transcrito de interesse, para as coletas CI e CV, respectivamente. Os arquivos são escritos em **find_peak.py** e usados em **compare_phase.py**.

O arquivo **arq_final.txt** contém informações sobre o período, fase, amplitude e pico ou vale em ZT18 para cada transcrito de interesse. Este consiste no arquivo final e objetivo do programa, escrito pelo algoritmo **compare_phase.py**.

O arquivo **read.me.txt** contém instruções e a descrição do programa.

O arquivo **test.txt** contém a lista de transcritos do Relógio circadiano expressos tanto em CI quanto em CV e pode ser utilizado para testar o programa.