

## Модуль спектрального анализа (Модуль Spectr)

### Назначение модуля

Модуль предназначен для обработки данных акселерометров или других датчиков за определенные интервалы времени для построения спектра колебаний и нахождения в полуавтоматическом режиме основных форм колебания и вычисления их параметров их пиков. Различает одиночные и двойные пики.

### Технические требования

Операционная система: Ubuntu 20/22, Linux OS 64 битная. PHP версии 7.3, модуль работы с PostgreSQL и ClickHouse.

### Требования к вычислительным ресурсам

1 ядро поддерживающее 64 битные вычисления, 500 Мбайт на жестком диске для временных данных.

### Описание работы

Структурно работу модуля можно поделить на две части: сбор, подготовку данных и отправку готовых данных, которую выполняет PHP скрипт «index.php». Обработка данных, формирование спектров и поиск основных форм колебаний выполняет программа RealTimeSpectrum. Модуль может быть вызван или из системы или из консоли в контейнере docker «php». Для запуска работы модуля запускается PHP скрипт «index.php». Блок-схема работы модуля представлена на рисунке.



Вначале работы скрипта «index.php» загружаются конфигурационные данные модуля из файла «config.php», содержащего данные к подключению к базам данных, он находится на директории выше модуля и файл конфигурации модуля «config.php», лежащего в директории модуля. Далее выбираются шины типа «an-d3» и «com-an-d3», шины для подключения акселерометров. В выбранных шинах выбираются устройства типа «and\_3». Далее просматриваются выбранные устройства и создается массив каналов этих устройств, у которых код равен «0.x» — ось X, «0.y» — ось Y, «0.z» — ось Z, «0.a» — модуль ускорения, обозначается через A.

serif;;#000000;;inherit>После формирования массива, обрабатываем оси. Проверяем, или есть настройка для каждого канала в конфигурационном файле модуля «config.php». Если данных настройки нет, по</font> <font inherit/inherit;;#000000;;inherit>обработка данной оси пропускается.</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Если канал сконфигурирован, то запрашивается данные по этому каналу в интервале «\$time\_interval» до времени запрошенного в начале запуска скрипта, переменная «\$timestamp».</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Далее проверяется и создаются папки для выходных файлов программы RealTimeSpectrum «/var/www/html/public/Spectr».</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Данные программой RealTimeSpectrum обрабатываются в двух вариантах. Первый — для полноценной спектральной мощности, данные записываются в директорию «/data», второй — для цветовой спектрограммы, данные записываются в директорию «/data\_partial». Для спектрограммы спектральная мощность строится с меньшим частотным разрешением для увеличения быстродействия. Далее данные записываются в директории с именем равным id устройства, далее id канала, после год, и последняя директория месяц.</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Для примера формирования пути, если id устройства 67, id канала 433, 2022 год и месяц сентябрь 9-й месяц, тогда путь будет выглядеть для первого случая «/var/www/html/public/Spectr/data/67/433/2022/09», для второго случая «/var/www/html/public/Spectr/data\_partial/67/433/2022/09».</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Далее формируются конфигурационные файлы «ofs\_options.ini» и «Config.path» для работы программы RealTimeSpectrum и записываются полученные данные в бинарном виде для обработки. Далее данные обрабатываются программой RealTimeSpectrum и полученные данные размещаются как указано выше, и полученные данные по пикам основных форм колебаний отправляются в базу данных. Обработка, формирование файлов и отправка данных выполняются для двух случаев подряд, так как для них используются одни и те же данные и должны использоваться одни и те же данные, что бы результаты были одинаковыми.</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Рисунок. Блок-схема модуля.</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Основные файлы модуля</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>- config.php - файл настроек программы обработки спектров, должен быть в gitignore. config\_example.php - пример файла настроек программы обработки спектров</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>-</font> <font inherit/inherit;;#000000;;inherit>Config.path - динамический файл настроек датчиков для работы программы обработки спектров RealTimeSpectrum, формируется в процессе работы модуля, должен быть в gitignore.</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>- ofs\_options.ini - динамический файл общих настроек работы программы обработки спектров</font> <font inherit/inherit;;#000000;;inherit>RealTimeSpectrum, формируется в процессе работы модуля, должен быть в gitignore.</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>- .gitignore - содержит список файлов и директорий, которые должны игнорироваться и не попадать в индексы</font> <font inherit/inherit;;#000000;;inherit>git</font> <font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>-</font> <font inherit/inherit;;#000000;;inherit>RealTimeSpectrum - программа для обработки данных датчиков и формирования спектров и поиска пиков

основных форм колебания.

```
index.php - PHP скрипт вызываемый для обработки спектров. Непосредственно в нем берутся данные из базы данных, подготавливаются для обработки и вызывается программа RealTimeSpectrum. Скрипт формируют пути, куда записывается спектр сформированный RealTimeSpectrum и отправляет результаты определения частот и декрементов колебаний в базу данных.
```

```
data_processing.sh - bash скрипт для запуска модуля через cron в контейнере.
```

```
source - директория с временными рабочими файлами модуля
```

```
index_check.php - вспомогательный PHP скрипт, позволяет проверить сколько в системе установлено устройств типа АНД-3
```

```
data_processing_test.sh - вспомогательный bash скрипт для запуска index_check.php в контейнере
```

## Программа RealTimeSpectrum

Программа RealTimeSpectrum обрабатывает данные сформированные акселерометрами и строит спектры колебания. Физический принцип, на котором основана работа программы, состоит в получении исходных данных ускорений с акселерометров от сборщика данных. Далее с помощью преобразования Фурье формируется спектральная мощность колебаний. Метод определения основных форм колебаний и их декрементов является полуавтоматическим. Для его работы необходимо задать интервал частот, в котором необходимо искать пик колебаний, и так же указать ищется ли одинарный или двойной пик. Далее специальным методом с помощью метода наименьших квадратов аппроксимируется пик или два пика в зависимости от настроек, и по данным аппроксимации вычисляются частота максимума пика и по ширине на полувысоте аппроксимированного пика — декремент колебаний.

## Настройка модуля

Настройка модуля производится в файле config.php, все параметры подробно описаны в config\_example.php, его содержание приведено ниже

```
$api_token = 'A8WQ0N4zYPQvxbVMvPVz2ZnhGGyfgjhbh'; api токен для доступа по rest api  
$time_interval = 40960; Интервал в секундах от текущего времени, за который запрашиваются данные с clickhouse  
$length_zap_full = 8192; Длина одной реализации для построения одного спектра  
$length_zap_spectrograms = 1024; Длина одной реализации для построения цветовой спектрограммы, так как вопрос ещё в производительности то длина уменьшена для быстрого рендера картинки  
$number_records_full = 2; Количество реализаций, по которым происходит усреднение для построения одного спектра  
$number_records_spectrograms = 16; Количество реализаций, по которым происходит усреднение для построения цветовой спектрограммы  
$basic_waveforms = 5; Количество основных форм колебания, которые обрабатываются скриптом PHP и отправляются в базу данных если данных на i-ю
```

форму нет после работы программы построения спектра, то она заполняется нулями

Настройка каналов для обработки В массив \$device\_code добавляется код устройства, к которому принадлежит канал в \$min\_frequency\_all и в \$max\_frequency\_all с ключом равным коду устройства указывается минимальная и максимальная граница интервалов, в которых нужно искать пики основных форм колебания. В \$double\_frequency\_all указывается количество пиков для поиска

```
$device_code=array('433', '434', '501', '502'); $min_frequency_all['433']='0.3 0.6 1.3';  
$max_frequency_all['433']='0.35 0.8 1.6'; $double_frequency_all['433']='1 1 1';  
$min_frequency_all['434']='0.3 0.6 1.3'; $max_frequency_all['434']='0.35 0.8 1.6';  
$double_frequency_all['434']='1 1 1'; $min_frequency_all['501']='0.3 0.6 1.3';  
$max_frequency_all['501']='0.35 0.8 1.6'; $double_frequency_all['501']='1 1 1';  
$min_frequency_all['502']='0.3 0.6 1.3'; $max_frequency_all['502']='0.35 0.8 1.6';  
$double_frequency_all['502']='1 1 1';
```

Более подробное описание что есть что

```
$device_code=array('433'); $device_code - набор кодов устройств, где устройствово
```

означает одну ось датчика, например 001X. Т.е. устройство с кодом 433 это датчик с номером 001, ось X. Для устройства 433 указываем интервалы, в которых нужно искать пики колебаний. Интервалы: \$min\_frequency\_all['433']='0.3 0.6 1.3'; \$max\_frequency\_all['433']='0.35 0.8 1.6'; Они означают 3 интервала, в которых искать пики. Интервал 1 0.3-0.35; интервал 2 0.6-0.8; интервал 3 1.3-1.6; И количество пиков, которые необходимо искать \$double\_frequency\_all['433']='1 1 1'; Здесь указаны три интервала, в них искать по одному пику. Варианты значений 1 либо 2. Файл для работы программы обработки спектров, общие настройки \$file\_options='number\_records = '.\$number\_records.' Количество реализаций, по которым будет строится аппроксимация спектра и находится параметры колебаний сооружения length\_zap = '.\$length\_zap.'

Длина одной реализации в количестве отсчетов, должно быть кратно 2<sup>n</sup>, 2 в степени n, где n целое число number\_of\_records = 30 Сколько в одном файле реализаций sampling\_time = 0.1 Время дискретизации датчиков в секундах time = «22\_01\_10\_23\_59\_43\_021» Время, с которого считывать данные, данные до этой временной ометки будут игнорироваться, если указать 00\_00\_00\_00\_00\_00\_000, то будут обрабатываться все данные time\_start = «2022\_02\_12\_07\_21\_00\_729» Время, с которого считывать данные, данные до этой временной ометки будут игнорироваться, если указать 00\_00\_00\_00\_00\_00\_000, то будут обрабатываться все данные time\_end = «.»\$year«. \_ckgedit\_QUOT».«\$month». \_ckgedit\_QUOT«.»\$day«. \_ckgedit\_QUOT».«\$hour». \_ckgedit\_QUOT«.»\$minute«. \_ckgedit\_QUOT».«\$second». \_000\_ckgedit\_QUOT\_big\_files = 0

Параметр, отвечающий за то, что мы берем большие файлы данных за сутки. 1 - мы берем, другое значение, не берем debug\_mode = 0 Режим отладки, выводятся промежуточные данные обработки, по умолчанию отключено processe\_all\_files = 0 Обработка всех файлов сразу, по умолчанию обрабатывается начиная с самого последнего rusian = 1 Включение русского языка, по умолчанию включен file\_size\_for\_processing = 1 Устанавливаем ограничение обработки файлов за один запуск time\_shift=0.245733333333333 Сдвиг по времени в секундах Пропуск времени между соседними спектрами при обработке всех файлов сразу, величина в секундах. detector\_option\_file = «Config.path» Имя файла с настройками для датчиков. fast\_processing = 0 file\_format=4 Формат файлов данных monitoring\_horizont=1; Файл для работы программы обработки спектров, настройка датчиков \$file\_options\_conf='{ limitation\_standart\_deviation=0 Среднеквадратичное отклонение limitation\_ejection=50

Выброс в процентах от диапазона measuring\_range=3600 Тип датчика (диапазон) AN3600, AN7200 axis='.\$detector\_name.' Номер датчика и его оси namefile=«'.dirname(FILE\_).'/source/» Относительный путь к данным savadata=«'. \$dir.'/» Относительный путь, куда сохранять результаты моделирования datatemperature=«'./» Относительный путь, откуда читать данные по температуре min\_frequency ='. \$min\_frequency.' Минимальная частота интервалов, в которых ищутся пики, в Гц, пример 0.4 0.8 2.3 max\_frequency ='. \$max\_frequency.' Максимальная частота интервалов, в которых ищутся пики, в Гц. Пример 0.6 1.1 2.5 double\_frequency ='. \$double\_frequency.' Количество пиков в интервале, допустимые значения 1 - один пик, 2 - два пика в интервале signal\_level=0; }';

## Настройка запуска по расписанию

Для запуска модуля из консоли нужно использовать скрипт «data\_processing.sh», который запускает модуль в контейнере docker «php». Для запуска модуля по расписанию, настраивается запуск скрипта «data\_processing.sh» через cron от имени root в требуемый интервал. При новой установке необходимо проверить права на запуск как исполняемой программы data\_processing.sh, RealTimeSpectrum. Содержание скрипта «data\_processing.sh» «/opt/monitoring/dc exec -T php php /var/www/html/SpectralAnalysis/Spectr/index.php» - Данные означают: - /opt/monitoring/dc - запуск docker - exec - команда выполнить - -T - не создавать виртуальное tty устройствово для виртуальной консоли. - php - запуск контейнера php - php /var/www/html/SpectralAnalysis/Spectr/index.php - запуск php скрипта index.php, путь указан в среде контейнер

## Тестирование и юнит тесты

<font 14px/Arial,Helvetica,sans-serif;;#000000;;inherit>Для тестирования можно запускать из консоли скрипт «data\_processing\_test.sh», который запускает в контейнере «php» скрипт «index\_check.php».</font>