Creating Grafana dashboards with live connection to Timescale wsprdaemon databases

New in this version: Specific to new release **Grafana V7.1.3** with its many changes. Also adds examples plots of derived variables in section 4 and use of plug-ins and non-time series graphs in section 5.

Contents

0. Installation and initial connection

Grafana is available for Windows, Mac, and Linux (including ARM) operating systems. Download and installation details are at:

https://grafana.com/grafana/download

There is no need to make any changes to the configuration options to run Grafana or to configure and install the dashboards covered in this note.

Connect to your own Grafana installation via a web browser to:

http://localhost:3000

Login with the default userid and password - both 'admin'. Then change your password.

If, however, you would like to try before you install your own version you are welcome to try the Grafana installation at http://logs1.wsprdaemon.org:3000
where you can log in as user Open and password Open
with access to two dashboards to illustrate Grafana's capability with WsprDaemon and WSPR data:

Open Access Noise Graph and Community SNR Comparisons.

The Open Access Noise Graph dashboard lets you choose the site_name, receiver_name, band_m and noise_type from the pull-down menus. You can also change the time interval from a pull-down list at top right. The site_names are of course only those stations using WsprDaemon to report their noise levels.

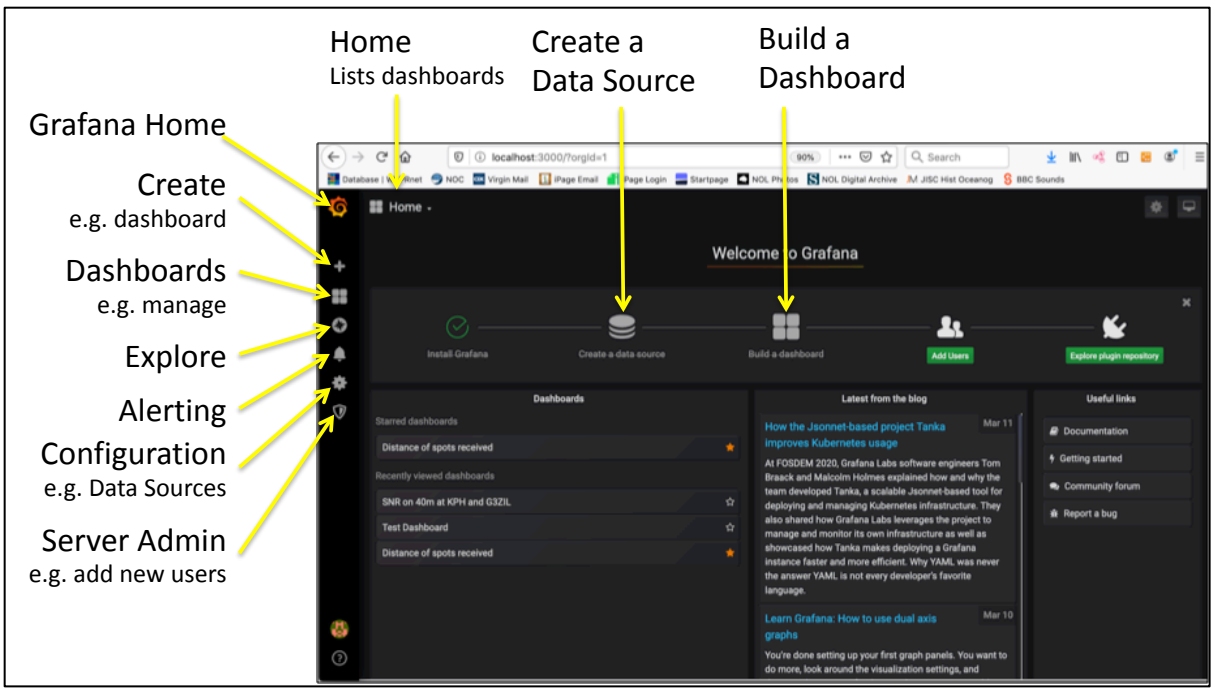
The Community SNR comparison dashboard lets you select any two stations reporting WSPR spots and a band (the list is limited to the bands reported by receiver_A). The panels give you a wealth of information on the SNR differences between the sites for the same stations heard at the same time.

WsprDaemon users are welcome to use the Grafana installation at
http://logs1.wsprdaemon.org:3000
where you can log in as user wdread and request a password (if not known to you) from gwyn at autonomousanalytics.com where you will have 'edit' rights on dashboards for you to see how they are constructed. You have permission to save your changes, or better, use save as; you can also create and save new dashboards. With this experience, you may then want your own installation.

The following notes are for those with their own installation, but sections 2 onwards are applicable to those with wdread access to at http://logs1.wsprdaemon.org:3000.

On login you should see the Home screen, below, showing you have installed Grafana, although in this example we already have a data source connected and some example Dashboards.

Several example Dashboards create here are available as Guide examples at logs1.wsprdaemon.org:3000 with userid Open and password Open.



1. Create a data source

From the screen above (click Grafana Home if you have explored some of the other screens), click on Configuration and then Data Sources. Click on Add data source. Scroll to PostgreSQL and select, bringing up the Settings page.

Give the data source a name, e.g. wsprdaemon_tutorial.

Enter the Host as logs1.wsprdaemon.org:5432

Enter Database as tutorial

Enter User as wdread and password as JTWSPR2008

For SSL Mode select disable

Leave Connection Limits as they are

Scroll down to the section PostgreSQL details (on my browser the scroll bar on the right is both very thin and a shade of grey just off-black making it hard to see).

For Version select 10 in the drop-down list (if 11 appears, select it instead).

Slide the TimescaleDB switch to the right. Leave Min time interval at 1m.

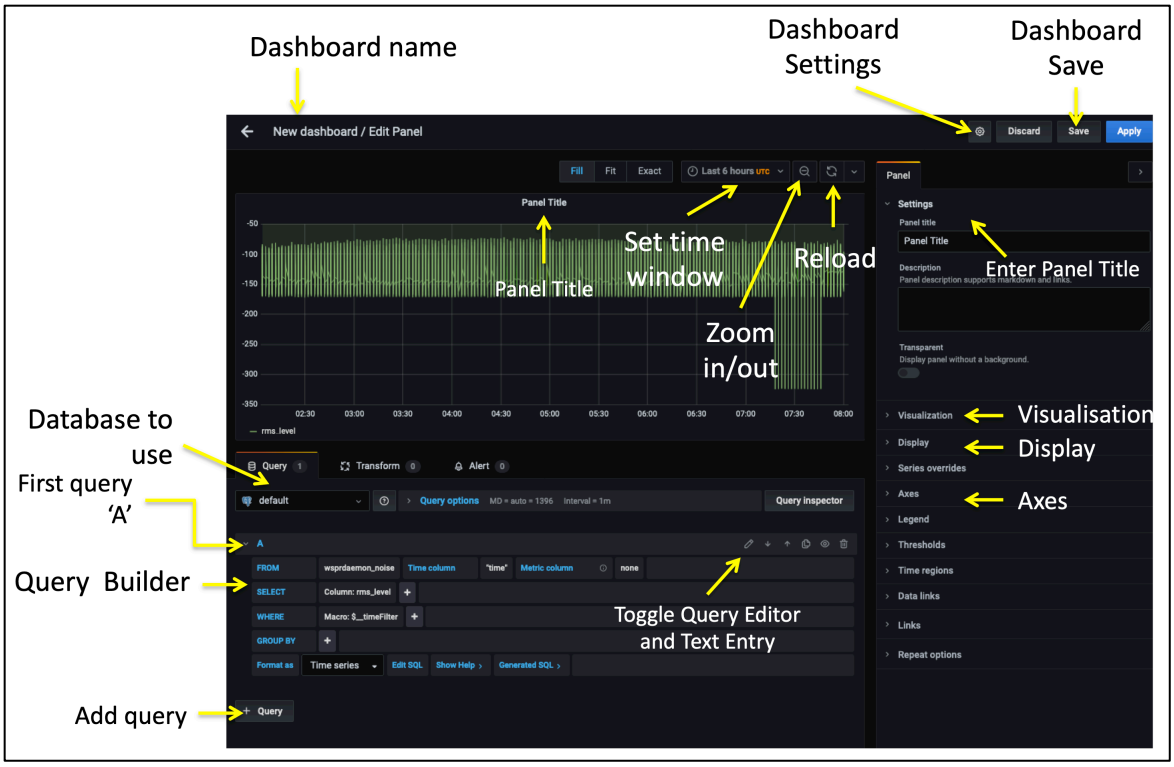
Scroll to bottom of the screen and click the green Save & Test button. All being well you should see a green banner with a tick and Database Connection OK message.

2. Create a dashboard

Hover over the Create icon at top left (big plus), click on Dashboard,

2.1 Create the first Panel.

Click on Add new panel, which will bring up a screen similar to that below.



The upper part of the screen shows a default time series graph - a Panel in Grafana jargon.

The lower part is a form-filling query builder to collect data from, in our case, a postgreSQL database with TimescaleDB extensions. At right (new in version 7) is an extensive list of option headings to name and then customise the graph. The important Save button is top right - too close to Discard for me!

The query builder is ready for Query 'A', others can be added later. As a first example, let us plot the SNR for OZ7IT at G3ZIL on 40m:

First, under Query, pull down the data source name e.g. wsprdaemon_tutorial.

On the FROM line click default and from pull-down options select wsprdaemon_spots.

Leave Time column as "time". After Metric column select tx_call from the pull down menu.

On the SELECT line, it may already show Column: "SNR", if it does not, click on whatever text is after Column: and select "SNR" from the pull-down.

If the WHERE line shows a Macro entry, click it until it shows remove, then click remove. Click on the + to the right of WHERE, in the empty box, click and you may see a pull down list, or you may not, type E and it may come up with the option Expression, if not type it. At this point a red error banner may come up, that's ok, we've not set values yet. Click on the first value and select rx_id (may not always 'take', if so try selecting again, if still nothing, type in 'G3ZIL', the single quotes are needed). Click on the plus again, and in the same manner, click and select in turn Expression: band = '40'. Click on the plus again, and in the same manner, click and select in turn Expression: tx_call = 'OZ7IT'.

At this point, data should appear. If there is no data, try zooming out in time using the zoom out icon (magnifying glass with -).

Save your work so far by clicking on the disk icon to Save Dashboard. Give the Dashboard a suitable name, e.g. SNR at G3ZIL, noting that we'll be adding further Panels to this Dashboard and so the Dashboard name should be appropriate.

2.2 Add a second (and subsequent) time series to this Panel.

After a save, we will have left the Query Builder, so click on the Panel Title then on Edit, which will bring back the Query Builder. We'd best give the panel a title, e.g. SNR of OZ7IT and N1ZPY, in the box Panel title under Settings at top right.

Click on the + Query button at bottom left, which will bring up a query builder for query 'B'. Of course, this second query should be along the same lines at the first, e.g. still SNR at G3ZIL on 40m but, say, for N8VIM. Repeat the steps above to build the query. Click Save.

2.3 Customise the Panel.

Click on the Visualisation heading on the right, you'll see a number of different types of graphs - for this example we will stick with 'Graph' the default. Close the Visualisation pull-down.

Click the Display pull-down, as you might want to use points rather than lines for this type of graph, if so, slide the lines switch to the left and the points switch to the right. For Point Radius you might set it to 1. Close the Display pull-down.

Click the Axes pull-down, under the Left Y column you might want to set fixed Y-min, and perhaps Y-max. You should also enter a Label for the Y-axis, e.g. SNR (dB in 2.5kHz). There's no need to alter other settings for this Panel.

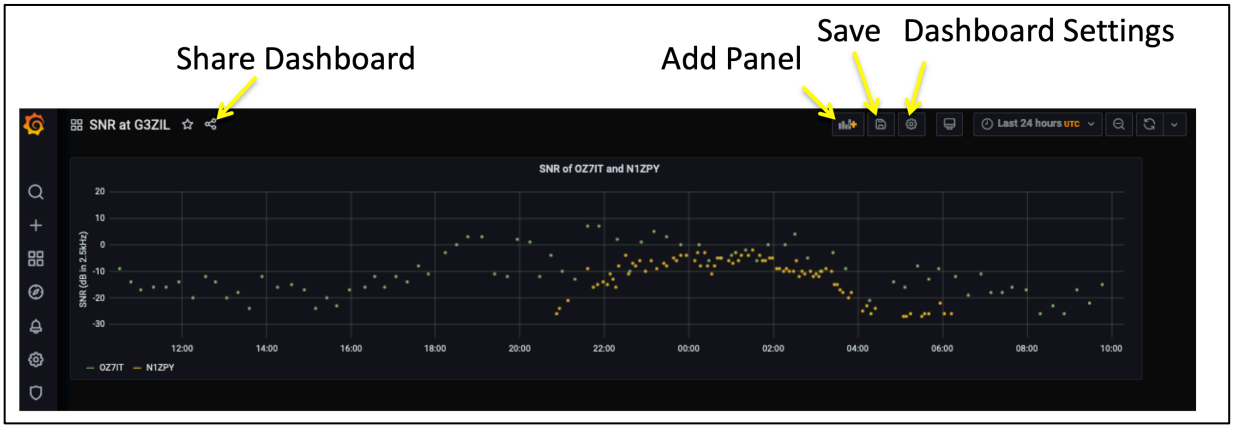
You can resize the panel by click dragging its bottom right corner.

Save the revised Panel.

2.4 Add a second Panel

Click on the Back Arrow at top left. It may leave you at the dasboard you've created or it may present you with a list, in which case select the one where you want to add a panel. This brings up a few other icons as in the screenshot below.

Click on Add Panel (the bargraph icon), then + Add new panel button, and the process is essentially as described for the first Panel. However, we can demonstrate here some additional graphical forms within Grafana, in this case a bar-graph grouped by time.



The theme is still 40 m at G3ZIL for the WHERE clause, but for this panel we will show the number of spots in each 10 minute interval.

Under Query select wsprdaemon_tutorial as the data source.

On the FROM line select wsprdaemon_spots.

On the SELECT line click on the + and you should see as an option Aggregate functions, and to its right, count. If you don't see Aggregate functions, type A and it should appear. Data should be seen in the new Panel at this stage. Column:"SNR" should already be present, and then Alias: "SNR" should automatically appear after count is selected, click on the "SNR" after Alias and over-type the proper descriptor, Spots in 10 minutes, which should then appear as the text alongside the key at the bottom left of the graph.

On the WHERE line, as for the first panel, remove the Macro, add Expressions for rx_id 'G3ZIL' and band '40'.

On the GROUP BY line click $-interval in brackets after time, and select 10m (i.e. 10 minutes).

Click the Display pull-down at right, and you may want to change to Bars rather than Lines..

On the Axes pull-down, for the Left Y axis enter a Label such as Spots in 10 minutes.

Top right, enter a Panel title such as Spots in 10 minutes.

Click Save.

Click on the back arrow at top left, selectthe Panel to resize by click drag on bottom right, adjust time range to suit. If you want to reorder the two panels, click and hold the title bar while you drag. Click the disk icon at top right to save.

2.5 Add a third Panel.

Grafana has a simple form of 'heat map' graph format. We can use it, for example, to show the variation with time of the spread of distance of received spots. Following the Add Panel procedure above for the FROM line, select wsprdaemon_spots. On the SELECT line click the Column variable and select 'km'. On the WHERE line click the Macro: option and select remove. Click the +, select Expression, then select 'rx_id' and then type in 'G3ZIL', click the +, select Expression, then select 'band' and then select or type in '40'. If there is an entry on the GROUP BY line, select the function and select remove.

On the Visualisation pull down at right select Heatmap. Close the Visualisation pull-down and select the Axes pull-down. You may want to set a suitable Y-min and Y-max. Under the Buckets column you may want to experiment to set the number of buckets for the x and y axes by number or by Size (e.g. in this case km for y, perhaps 500, and perhaps 20m for x - 20 minutes). Close the Axes pull-down and open the Display pull-down. Under Scheme - try different colour scales to use. And on the Color scale boxes set appropriate min and max, or leave as auto. To right of Legend select Show Legend option for a scale bar. Close Display and open Settings to give the Panel a title. Click Save.

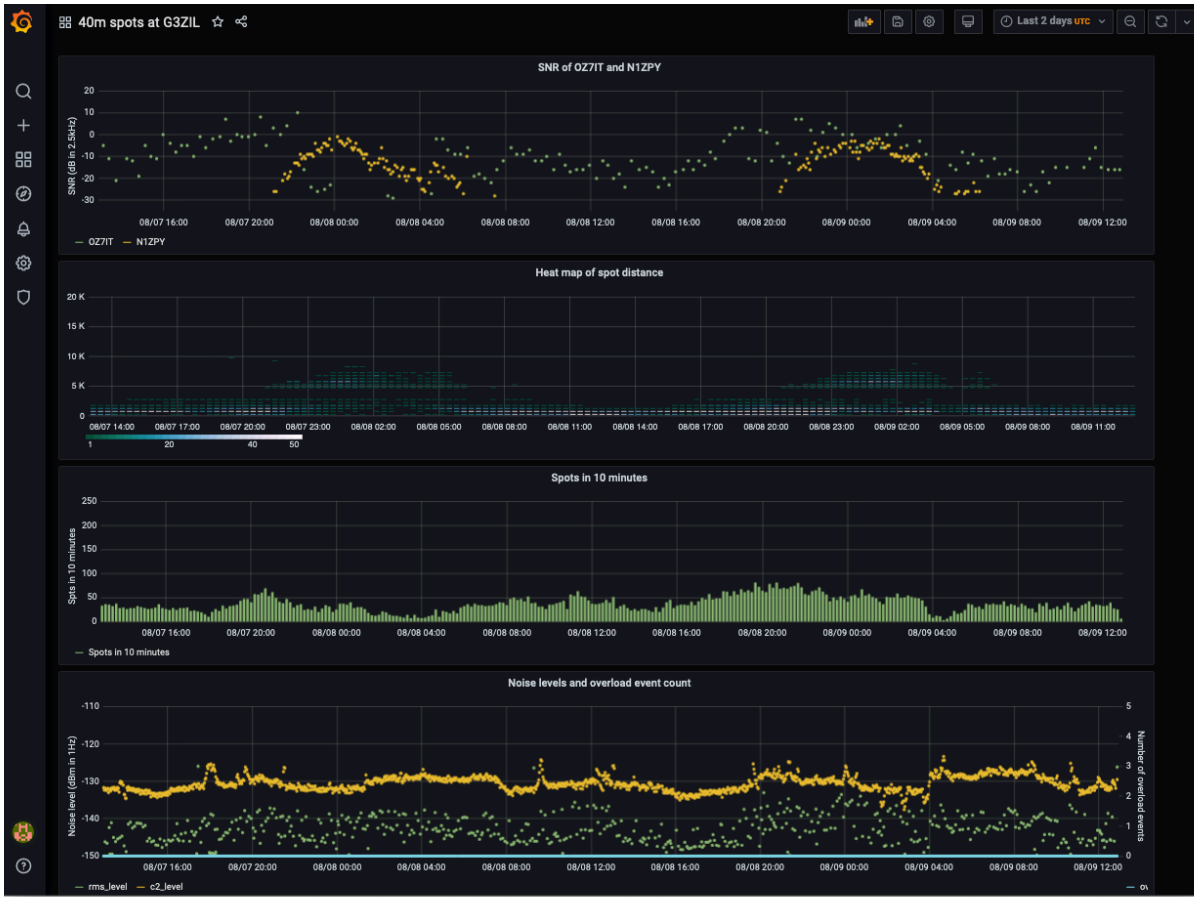
2.6 Add a fourth Panel

From wsprdaemon version 2.8a RMS and c2_FFT noise estimates are sent to the Timescale database on the wsprnet.org server. Noise graphs can therefore be presented alongside spots data. Following the steps as above, but on the FROM line select wsprdaemon_noise as the table. On the SELECT line, Column: rms_level may already be present, click +, select Column and then select c2_level. We can also show the overload flag instances, add Column: ov. On the WHERE line click Macro: and select remove, click +, then Expression, band and '40', click +, then Expression, site, 'G3ZIL'.

Open the Display pull-down at right, under Draw Modes, Points are best, and because of the 2-minute interval data, select Point Radius 1. Close the Display pull-down and open Axes, you will want to set limits on the Left Y axis, and add a Label e.g. Noise Level (dBm in

1Hz). Click on the line next to ov on the graph legend at left just under the plot. Click on the Y-axis tab and move the slider to the right to use the right-hand axis for the overload values. On the Axes pull down on the right, set Decimals to 0 and add a Label such as Number of overload events. Close the Axes pull-down, open Settings and give the Panel a title. Click the back arrow at upper left, resize the panels if need be, rearrange them, click the disk icon top right to save.

The Dashboard we have created should look like the screenshot below. Having gone through this tutorial for three different graph forms and using both left and right Y axes you should now have enough working knowledge to create your own simple Panels and Dashboards.



2.7 Change Dashboard Settings

Click the Dashboard Settings (cog) icon at the top of the screen.

Under Time Options you can select the Timezone to be determined by the local browser time or UTC. I suggest using UTC.

For Auto-refresh set a value such as 10m.

For Now delay now set a value of 1m to ignore last minute, which may have partial data.

Click Save on the left. You could also click Save As to use this Dashboard as a template for others, e.g. for different bands.

Click on the back arrow to get back to the dashboard.

3. Creating a Panel with pull-down selections

The Dashboard and panels described above have had the variable field selections to plot 'hard-coded' into the Query Builder and hence the plots themselves. Using the Template and

Variables features in Grafana we can construct panels with pull-down selections to enable general-purpose plots, along the lines of the original Grafana noise plots produced by Tommy Nourse, KI6NKO. Our example will be a noise measurements panel.

3.1 Pull-down selections within the WHERE clause

On opening Grafana click the + at left and select (to create a) Dashboard. Click on + Add new panel. Click the Dashboard settings cog icon at top right, give the Dashboard a name; in Time options set Timezone to UTC, Auto refresh to 10m and Now delay to 1m.

On the menu list at top left click on Variables, click on Add Variable. For the noise example we need three variables: Site, Receiver and Band, we'll call then site_name, receiver_name and band_m. Under the General heading, enter the Name - site_name. Under Query options, Data source, select wsprdaemon_tutorial, for Refresh, select On Dashboard Load.

The next step is to code the SQL query that will generate the list of site names for the pull-down list. In the Query box enter:

```
select distinct site from wsprdaemon_noise;
```

From the noise table this selects the distinct (unique) site names.

Set Sort as Alphabetical (asc), move the Multi-value slider to the right, to select more than one site to plot. You'll see a preview list of unique site names. Click Add then Save. Click on Duplicate at right against the site_name entry. Click on the copy, rename to receiver_name.

Edit the Query box to read:

```
select distinct receiver from wsprdaemon_noise where site in ($site_name);
```

As well as selecting distinct receiver names, we are specifying only those receiver names applicable to the sites we have already selected[1]. Check the Multi-value slider is to the right. You may well see error messages in a red box, ignore them. Click Update.

Click on the large type Variables at the top. Click Duplicate on the receiver_name line, click on the copy line, as the Variables>Edit page comes up, change the Name to band_m and edit the query box to read:

```
select distinct band from wsprdaemon_noise where site in ($site_name);
```

Set Sort order to Numerical (asc). Click Save dashboard at left. Click the Go Back left arrow, top left. Click the Panel Title and select Edit, which will bring up the Query builder. Select wsprdaemon_tutorial as data source. For Metric pull-down select receiver. On the Select line click the + and select Column, click to select c2_level if a second rms_level has appeared. On the WHERE line click Macro and remove, click the + then select Expression, pull down the first value to be site, and for the second value, select $site_name (i.e. the variable, and not a fixed name). Click + again for Expression, receiver, $receiver_name, + again, for Expression, band, $band_m. Error messages may appear, ignore for now.

In the Format as line, click on Edit SQL, the SQL may well look like the following:

```
SELECT
  "time" AS "time",
  receiver AS metric,
  rms_level,
  c2_level
FROM wsprdaemon_noise
WHERE
  site = '$site_name' AND
```

[1] This helps minimise confusion as there are many receiver names in the list with varying styles.

```
  receiver = $receiver_name AND
  band = $band_m
ORDER BY 1,2
```

There is a problem here - the Query Builder hasn't formed the query correctly for postgreSQL, the use of "=" in the WHERE clause is appropriate for a specific site, receiver or band, but not for a list. For a list the correct syntax is WHERE site IN ($site_name); so IN instead of =, and parentheses instead of single quotes.

In addition, we can expand the metric so that the legend has site, receiver name and band. So edit the SQL to read:

```
SELECT
  "time" AS "time",
  (site, receiver, band) AS metric,
  rms_level,
  c2_level
FROM wsprdaemon_noise
WHERE
  site IN ($site_name) AND
  receiver IN ($receiver_name) AND
  band IN ($band_m)
ORDER BY 1,2
```

We should now see the three pull downs properly populated with options and some data graphed. Open the Display pull-down at right, select points rather than lines, and set Point Radius to 1. Close Display, open Axes, add Left Y axis label: Noise level (dBm in 1Hz).

Close Axes, open Settings, give the Panel a title, you can refer to the variables, e.g. :

Noise level at $site_name for $receiver_name on $band_m m

Click the disk save icon at top, saving current time range and current variables.

If your expected pull down options do not appear, click on the graph panel, or the refresh icon at far top right and wait a few seconds. Here is the finished panel displaying data from OE9GHV and G3ZIL on 40m for two days.

3.2 Pull-down selections within the SELECT clause

We can also create pull-down selection lists for a choice of column names, for example to select c2_level or rms_level or both from the wsprdaemon_noise table. In this case we need the variable for the selection to be accessible in the SELECT clause within the Query Builder. This is rather more complicated and not at all intuitive.

Continuing with the example wsprdaemon_noise table as in 3.1, follow the process to create a new Variable as in that section; let us call the variable noise_type. In Query Options, set

Data source as before, Refresh is On Dashboard Load, and in the Query line use the following to list the column names:

SELECT json_object_keys (to_json ((SELECT t FROM public.wsprdaemon_noise t LIMIT 1)));

Explanation: The inner SELECT t FROM public.wsprdaemon_noise t LIMIT 1 returns one line of data from the table, the to_json provides it in a form for json_object_keys to list the keys (column names) of the data.

Alone, this will list us all the column names (see the Preview of values at bottom), but we just want c2_level and rms_level, so in the Regex box enter

/level/

to only list the output from the SELECT that includes 'level'[2]. Select Sort as alphabetical (ascending). In Selection Options slide Multi-value on. Save the Variables and then go back to the Dashboard and pull down edit from its title so we can return to the Query Builder and look at the SQL.

The SELECT clause in the Query Builder needs to be changed to make use of the variable noise_type. In needs to change to the following, **note** the curly braces in ${noise_type:csv}:

```
SELECT
  "time" AS "time",
  (site, receiver, band) AS metric,
  ${noise_type:csv} as value
FROM wsprdaemon_noise
WHERE
  site in ($site_name) AND
  receiver IN ($receiver_name) AND
  band IN ($band_m)
ORDER BY 1,2
```

It took me quite some time to find the correct syntax for ${noise_type:csv}. The curly braces and :csv signify "disable quoting" and accepts a comma separated list as its input.

3.3 Queries using SQL, not the Query Builder

There are queries we may want to use that are not supported by the Query Builder options. Grafana allows direct entry of all of the SQL statements needed to form a query. In this example we have three direct entry queries to form a panel to show spot distance statistics in 10 minute intervals.

Use what you've learnt above to create a new Dashboard and a panel, this time with wsprnet as the data source so we can look at all reporters not just WsprDaemon users.

First set up Variables, as we will use this Dashboard to compare two Reporters and two bands we need four variables:

receiver_A with query select distinct "Reporter" from spots;

band_A select distinct wd_band from spots where "Reporter"='$receiver_A';

and repeat for _B

[2] See https://www.boost.org/doc/libs/1_38_0/libs/regex/doc/html/boost_regex/syntax/basic_extended.html for a guide to regular expression syntax

In the Settings pull-down for the first panel the title can be Distance statistics for $receiver_A on band $band_A and we will create a second panel for receiver_B on band_B.

The statistics we will use are the median, lower quartile and upper quartile, where the following SQL is entered in queries A, B and C:

```
SELECT
  $__timeGroupAlias("wd_time",10m),
    percentile_disc(0.5) within group(order by spots.distance) as
"""median"""
FROM spots
WHERE
  $__timeFilter("wd_time") AND
  "Reporter" = '$receiver_A' AND
  wd_band = '$band_A'
GROUP BY 1
ORDER BY 1
```

Note: it is quite possible that you may see an error: pq: could not find pathkey item to sort, I have yet to find the root cause of this - it is not an error in the SQL, the work-around is to ask for longer data intervals, e.g. if it happens with one day selected, try 2, then 3 etc.

click + Query then insert

```
SELECT
  $__timeGroupAlias("wd_time",10m),
    percentile_disc(0.25) within group(order by spots.distance) as """lower
quartile"""
FROM spots
WHERE
  $__timeFilter("wd_time") AND
  "Reporter" = '$receiver_A' AND
  wd_band = '$band_A'
GROUP BY 1
ORDER BY 1
```

click + Query then insert

```
SELECT
  $__timeGroupAlias("wd_time",10m),
    percentile_disc(0.75) within group(order by spots.distance) as """upper
quartile"""
FROM spots
WHERE
  $__timeFilter("wd_time") AND
  "Reporter" = '$receiver_A' AND
  wd_band = '$band_A'
GROUP BY 1
ORDER BY 1
```

Adjust the plot using the Display pull-down to change to Points with Point Radius 1 rather than lines. Click Save. **Note:** This panel will take a fair time to open as it is getting pull down list options for all wsprnet reporters.

Click the bar graph Add Panel icon top right, and repeat the above steps but refering to _B rather than _A, remembering to set the data source.

Resize the panels to suit, and reorder so -A on top.

An example Community HF Distance Stats dashboard is shown below, where receiver_A has been selected as WA2ZKD (New York State) and receiver_B as AI6VN/KH6 on Maui, and both bands are 20m - an example of where we want to compare the statistics for two different stations on the same band, here over 48 hours. The lower plot shows an example where we compare the statistics for the same station on two bands, here 20 and 40m.

Note that variables - used for the pull-down selections - are defined for a Dashboard, not individual panels. Therefore, the list is above Panel A, while Panel A only uses the first two variables and Panel B the second two.

4. Time series graphs of derived variables

As TimescaleDB is built upon postgreSQL, a full-featured relational database, we can derive variables from data of two (or more) reporters to plot with Grafana.

4.1 SNR comparison between two reporters

In this example we will form the SNR difference for a sender received by two reporters, if that spot is on the same band and at the same time for both reporters. The Dashboard built in this example can be extended with other panels as described in section 3, e.g. with heatmaps of distance and azimuth at the receiver.

Create a new Dashboard and add a new panel. We need three variables - receiver_A, receiver_B and band_m, create them via the Dashboard settings icon as in section 3.1 using wsprnet as the data source and spots as the table, and in the meantime give the Dashboard a name, set timezone as UTC etc.

Select wsprnet as the data source. In the Edit SQL box enter the following:

```
SELECT
  s1."Date" AS "time",
  (s1."dB" - s2."dB" ) as "dB_difference"
from spots s1
inner join spots s2 on s1.wd_time = s2.wd_time
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
```

In the SELECT line we form the difference of the SNR for the two selected receivers. s1 and s2 are both aliases for the table spots - declared in the FROM line and the INNER JOIN line. Two aliases for the same table result in a "self-join" query, as if we had two tables. The JOIN happens when the time, CallSign received and band are the same for the two reporters.

Via the Settings pull-down enter a panel title, e.g.

SNR comparison between $receiver_A and $receiver_B on $band_m m

Via the Display pull-down select Points and Point Radius 1, and via the Axes pull-down set appropriate Left Y axis min and max and give the axis a label. An excellent suggestion from Glenn N6GN was to add a line at zero. This can be done using the Thresholds pull-down, select Add threshold, set T1 as gt (greater than) and enter 0 for the threshold, and slide the Fill button to the left - no fill.

The scatterplot is useful, but we can add a panel with median and quartiles that does help.

Click the back arrow at top left and the bargraph Add panel icon. Select wsprnet as the data source. Into query A Edit SQL box enter:

```
SELECT
  $__timeGroupAlias(s1.wd_time,20m), percentile_cont(0.25) within
group(order by (s1."dB" - s2."dB")) as """lower quartile"""
FROM spots s1
join spots s2 on s1.wd_time = s2.wd_time
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
WHERE $__timeFilter(s1.wd_time)
GROUP BY 1
ORDER BY 1
```

Click + Query and enter into the Edit SQL box

```
SELECT
  $__timeGroupAlias(s1.wd_time,20m), percentile_cont(0.5) within
group(order by (s1."dB" - s2."dB")) as """median"""
FROM spots s1
join spots s2 on s1.wd_time = s2.wd_time
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
WHERE $__timeFilter(s1.wd_time)
GROUP BY 1
ORDER BY 1
```

Click + Query and enter into the Edit SQL box

```
SELECT
  $__timeGroupAlias(s1.wd_time,20m), percentile_cont(0.75) within
group(order by (s1."dB" - s2."dB")) as """upper quartile"""
FROM spots s1
join spots s2 on s1.wd_time = s2.wd_time
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
WHERE $__timeFilter(s1.wd_time)
GROUP BY 1
ORDER BY 1
```

Use the Settings, Display and Axes pull-downs to customise, although you may want to have lines for this plot, but set area fill to 0, and have Points. Using the Threshold pull-down add a line at zero as above.

The screenshot below shows the resulting Dashboard, here for G3ZIL and G4HZX on 40m; despite a great deal of variability there is a clear repeatable daily pattern in the statistics.

5. Non-time series graphs

While it may seem a little odd, given we have specifically chosen a time series database, there are instances where we would like a variable other than time on the x axis, e.g. distance. As installed, Grafana has no such capability, but there is a range of third-party plug-ins that do. I have installed the plotly plug-in[3] on logs1.wsprdaemon.org.

5.1 SNR with distance

While this may not be the most useful plot, or straightforward to interpret, it serves as a simple exampe of a non-time series plot and the use of plotly. Create a new Dashboard and + Add panel as before. In the Visualization pull-down scroll down to Plotly, and select. You will see that the list of pull down options on the right is different to what we have seen before.

Select Dashboard settings at top right, as before give the Dashboard a name etc. Create two Variables for the receiver and band, with wsprnet as the data source, as before.

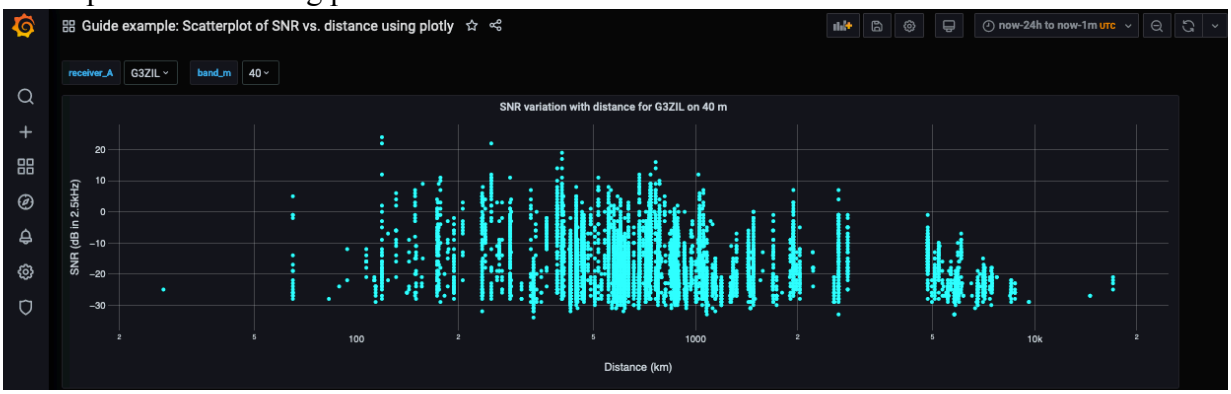
Edit the Panel, select wsprnet as the data source and enter the following into the Edit SQL box:

```
SELECT
  "wd_time" AS "time",distance,
  "dB"
FROM spots
WHERE
  $__timeFilter("wd_time") AND
  "Reporter" = '$receiver_A' AND
  wd_band = '$band_m'
ORDER BY 1
```

Select the Display pull-down on the right, enter an X axis title, select type as Linear, but you might also want to have a look at Log, enter a Y axis title. In the Traces pull-down, enter a Name for the trace, under the Metrics heading select distance for the X axis and dB for the Y axis. Under the Markers heading a size of 5 seems sensible and colour as Solid. In the Solid

[3] See https://grafana.com/grafana/plugins/natel-plotly-panel for details

box you can type a colour by name, e.g. cyan. Under Text you can select Metric as distance and Show as Hov, which will show the distance and SNR when you hover over a spot on the scatterplot. The resulting panel is shown below



5.2 3D SNR Difference with distance and with azimuth at the receiver with pull-downs

The plotly plug-in for Grafana has few options, but one that may possibly be useful is the 3D scatterplot, although it may also be a bit of a gimmick. This Dashboard starts out as for 5.1 and 5.2 but the SQL code below adds a third variable in the select, so we have azimuth at the receiver, distance and the calculated SNR difference.

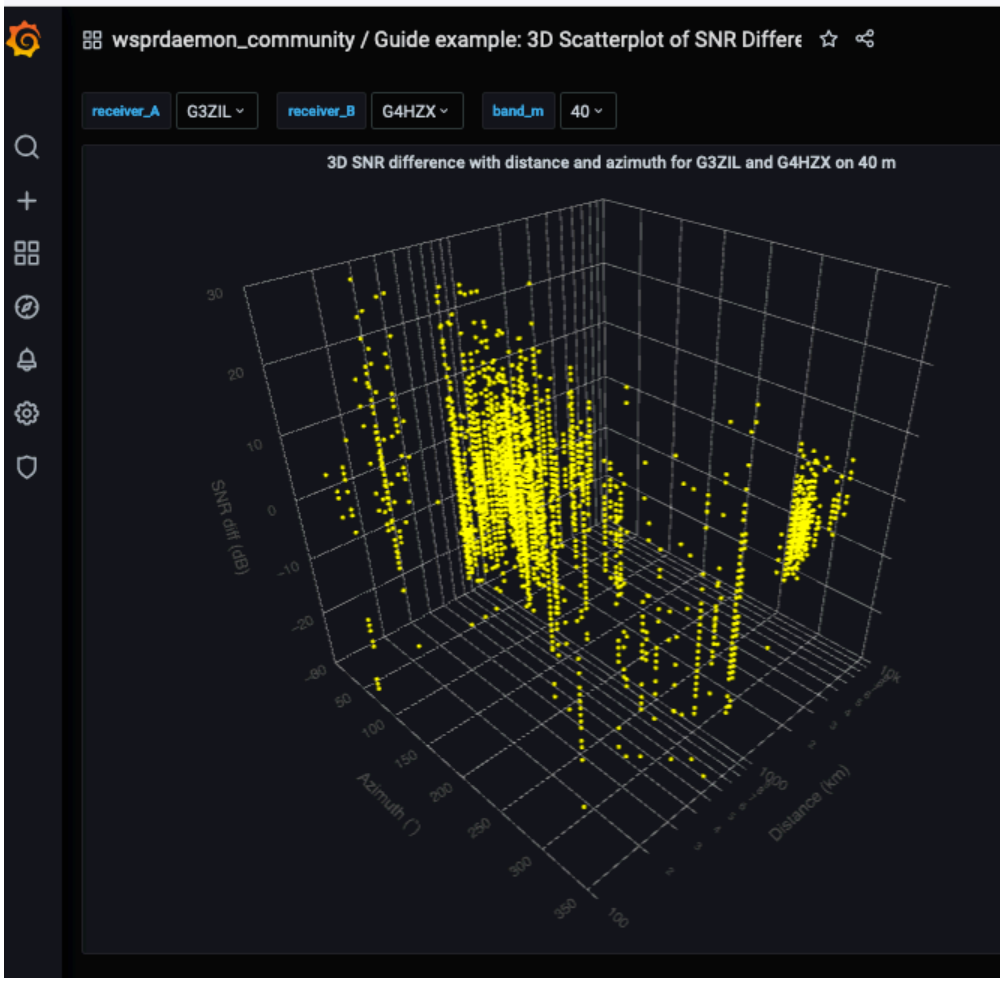
In the Display pull-down, under Options select Type as Scatter (3d), and Drag as pan, the latter lets us use a mouse to rotate the final 3D data plot. Under the X axis heading set a title that will match the x axis we will choose in the Traces pull-down, e.g. Distance (km), Type is perhaps best as Log, Range as Between and Min and Max as powers of 10, so 4 is 10,000 and 2 is 100. Although the labels are Min and Max, a more suitable axis is given if you reverse them, in this case I set 4 for Min and 2 for Max that makes the axis values increase from left to right. Under Y axis, which will be Azimuth, give an appropriate title, Type as Linear, Range as Between and 0 and 360 as Min and Max. Under Z axis, which will be SNR difference, use Type as Linear, Range as Between and set appropriate Min and Max.

In the Traces pull-down, give the Trace a name. Under Metrics, set X Axis to km, Y axis to azi and Z axis to db_difference. Under Markers check Show is to the right, Symbol is Circle, Size is 2, Colour is Solid, and Solid is a colour name of your choice. Under Lines, Show should be off, under Text, select Metric as km and Show to Hov, as we hover over points we can read the distance, azimuth and SNR difference.

This is the SQL code to insert into the Edit SQL box:

```
SELECT
  s1."wd_time" AS "time",
  s1.distance as km,
  s1.wd_rx_az as azi,
  (s1."dB" - s2."dB") as db_difference
FROM spots s1
join spots s2 on s1.wd_time = s2.wd_time
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
WHERE
  $__timeFilter(s1."wd_time",10m)
ORDER BY 1
```

Here is an example of the resulting Dashboard panel:



5.3 2D SNR Difference with distance and with azimuth at the receiver with pull-downs

Dear reader, you may well have had enough of hand-holding explanation by now. This Dashboard is a variant on the 3D representation, create the Dashboard and panel as before, use the options on the right for a 2D scatterplot, use the top SQL for the distance panel, duplicate it, and use the SQL beneath for the azimuth panel.

```
SELECT
  s1."wd_time" AS "time",
  s1.distance as km,
  (s1."dB" - s2."dB") as db_difference
FROM spots s1
join spots s2 on s1.wd_time = s2.wd_time
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
WHERE
  $__timeFilter(s1."wd_time",10m)
ORDER BY 1
```

and for azimuth

```
SELECT
  s1."wd_time" AS "time",
  s1.wd_rx_az as azi,
  (s1."dB" - s2."dB") as db_difference
FROM spots s1
join spots s2 on s1.wd_time = s2.wd_time
```

```
and s1."CallSign" = s2."CallSign"
and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
and s1.wd_band='$band_m' and s2.wd_band='$band_m'
WHERE
  $__timeFilter(s1."wd_time",10m)
ORDER BY 1
```

These two panels result in the Dashboard below.

5.4 Scatterplots with distance, azimuth and time of the SNR at one station for spots not heard on the same time and same band as another station

This is a nice example of what can be done with a relational database. As in section 5.3 I will skip the initial set-up of a Dashboard with a plotly panel, concentrating on the SQL to enter into the Edit SQL box. The data source for all is wsprnet, so we can access all reporters. The bulk of the SQL is the same for the three panels (two using plotly and one using a time series Graph), that for the first panel, distance, being:

```
SELECT
  s1."wd_time" AS "time",
  s1.distance as km,
  s1."dB"
FROM spots s1
left join spots s2
on s1.wd_time = s2.wd_time
  and s1."Reporter" = '$receiver_A' and s2."Reporter" = '$receiver_B'
  and s1.wd_band='$band_m' and s2.wd_band='$band_m'
  and s1."CallSign" = s2."CallSign"
where s1."Reporter" = '$receiver_A' and s1.wd_band='$band_m' and
s2."CallSign" is null
and s1.wd_time between $__timeFrom() and $__timeTo()
```

The main aspects are:

- The SELECT clause accesses wd_time, distance (for the X axis) and SNR here as "dB" (for the Y axis).

- In the FROM clause we alias spots as s1 and use a left join with alias s2 of the same table spots with our list of criteria: wd_time, reporters as those selected in the pull-downs, ensuring band is the same for both, and match the sender CallSigns. A left join can be thought of as the area to the left of an intersection on a Venn diagram.
- It is the WHERE clause that finds the spots we want - where we have a receiver_A and the selected band but no entry for the CallSign for receiver_B for wd_time between $__timeFrom() and $__timeTo(), which are Grafana global variables for the from and to times in the time picker at top right of the Dashboard.

The SELECT clause for the other two panels being, for azimuth:

```
SELECT
  s1."wd_time" AS "time",
  s1.wd_rx_az as azi,
  s1."dB"
```

and as a time series:

```
SELECT
  s1."wd_time" AS "time",
  s1."dB"
```

The resulting Dashboard is shown below for spots heard on 30m by G3ZIL, using an N6GN Active Antenna tilted 22˚ from the horizontal, the down side pointing NE, and G3ZIL/S using a pair of 40m dipoles.

6. Hourly Heatmaps

To be added in next version.