\*\*Documentation for the Python program "MVF\_Q\_withMaskeddataDF\_Proto"

The `MVF` function is designed to analyze and visualize the sensitivity of a cost model related to battery performance. Here are the details regarding its usage and functionality:

1. \*\*Function Parameters\*\*:

- `Var1`: The first variable (e.g., 'Concentration\_pos').

- `min1`, `max1`, `steps1`: Parameters for generating an array of values for `Var1`.

- `scale1`: Scaling option for `Var1` (either "Exponential" or "Linear").

- `Var2`: The second variable (e.g., 'Current\_Density').

- `min2`, `max2`, `steps2`: Parameters for generating an array of values for `Var2`.

- `scale2`: Scaling option for `Var2` (either "Exponential" or "Linear").

2. \*\*How It Works\*\*:

- The function generates arrays (`array1` and `array2`) based on the specified parameters.

- It initializes an output matrix (`Output`) with zeros.

- For each combination of values from `array1` and `array2`, it calculates the sensitivity value using the `SVF` function.

- If certain conditions are met (e.g., `eG16` or `pH10` are negative), the sensitivity value is set to `None`.

- The sensitivity plot is generated using `sensitivity\_plot.plot\_sense`.

- The output matrix is converted to a Pandas DataFrame and saved to an Excel file.

3. \*\*Example Usage\*\*:

```python

MVF(

'Concentration\_pos',

0.01,

4,

80,

'Exponential',

'Current\_Density',

1,

500,

300,

'Exponential'

)

```

4. \*\*Notes\*\*:

- Ensure that the required libraries (`numpy`, `pandas`, and `sensitivity\_plot`) are installed.

- Customize the variable names, ranges, and scaling options according to your specific use case.