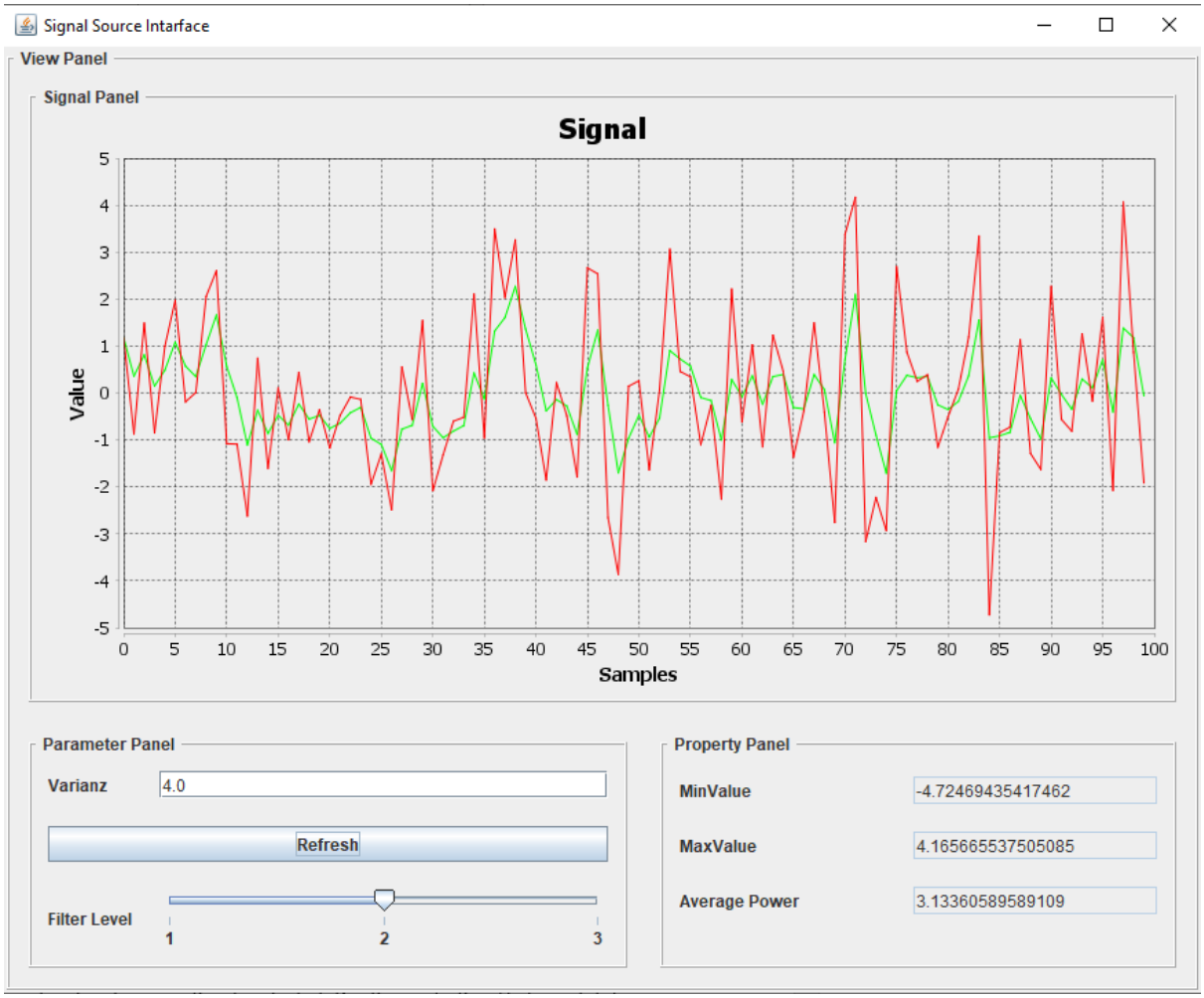
Übung: Anwendung einer Schnittstelle zum Datenaustausch

Ziel dieser Übung ist es, die Kommunikation zwischen Klassen mittels einer Schnittstelle zu demonstrieren. Dies geschieht am Beispiel eines Signalgenerators, der völlig autark ein Zufallssignal generiert. Über die Schnittstelle **DataListener**, welche von einem beliebigen Model implementiert wird, kann dann spezifisch das Signal ausgewertet werden, ohne dass der Signalgenerator etwas von dem Model weiss.



Die Gesamtapplikation ist im **Model-View-Controller** Entwurfsmuster gehalten. Im **ParameterPanel** wird die Varianz des Zufallssignals vorgegeben, was durch Betätigung von **Refresh** berechnet wird. Mit dem **JSlider** Filter kann die Filterung des Signals (grüne Kurve) in drei Stufen vorgegeben werden. Im **PropertyPanel** werden einige Eigenschaften des Signals dargestellt. Das **PlotPanel** ist bereits vorgegeben und verwendet zur Darstellung der Signale das Package **jfreechart**, was kostenlos unter https://www.jfree.org/jfreechart/ verfügbar ist.

Alle Panels der **View** sind im GridbagLayout organisiert und sollten in etwas die Gestalt wie auf der Skizze oben annehmen. Das genau Verhalten bei Vergrösserung dürfen Sie frei nach Ihrem Geschmack implementieren.

Das Aufdatieren der Modeldaten in der View erfolgt mittels **Observer-Observable Mechanismus**.

Aufgaben :

a) Machen Sie sich mit dem Klassendiagramm vertraut und identifizieren Sie die typischen Strukturen des MVC Patterns. Eine Besonderheit ist die Verbindung des Models mit der Signalquelle mittels der Schnittstelle **DataListener**. Das Grundgerüst des Klassendiagramms ist in der Vorlage bereits implementiert. Weitere Hinweise finden Sie auch als Pseudocode.

b) Realisieren Sie in der **init()** Methode der Klasse **SignalSourceApplikation** die einzelnen Schritte zur Erzeugung der MVC Struktur. Gemäss Klassendiagramm werden die Objekte lokal in **init()** deklariert und erzeugt. Fügen Sie die erzeugte **view** dem Frame hinzu.

c) Implementieren Sie zunächst die Klasse **View** und alle **Panels**. Die Aufgaben der einzelnen Methoden sollten im Kontext des MVC-Patterns klar sein. Das **Plotpanel** ist vollständig vorgegeben und realisiert die Darstellung der Signale mittels der Klasse **JFreeChart**

d) Nun können Sie die Steuerung in der Klasse **Controller** realisieren. Weitere Informationen sind als Kommentar im Code zu finden.

e) Die Klasse **Model** hält das Signal und die berechneten Eigenschaften. Implementieren Sie die Methoden der Klasse und beachten Sie die notwendigen Abläufe aufgrund des verwendeten MVC Pattern. Die Methoden sind weitestgehend selbsterklärend, weitere Infos sind im Kommentar und im Hinweis unten zu finden.

f) Implementieren Sie nun noch die eigentliche Signalquelle. Das Zufallssignal wird mit Hilfe der Klasse **Random** wie im Code beschrieben erzeugt. Machen Sie sich den Datenaustausch zwischen Model und SignalSource bewusst und beschreiben Sie es in eigenen Worten.

Hinweise:

- mittlere Leistung

$$meanPower = \frac{1}{N}\sum_{n=0}^{N-1} y[n]^2$$

- Berechnung des gefilterten Signals *y* aus Rohsignal *x* mit einem PT1 Filter mit Filterkonstante FK:

$$y_n = FK \cdot y_{n-1} + (1 - FK) \cdot x_n$$

Klassendiagramm: