FIR Filter Coefficient Generator

A comprehensive GUI application for designing FIR (Finite Impulse Response) filters with real-time visualization and coefficient quantization capabilities.

Features

Filter Design

- **Filter Types**: Low Pass, High Pass, Band Pass, Band Stop
- **Configurable Parameters**:
  - Number of coefficients (automatically adjusted to odd numbers)
  - Cutoff frequencies (normalized 0-0.5 range)
  - Sampling frequency
- **Real-time Visualization**:
  - Magnitude response in dB
  - Phase response
  - Cutoff frequency indicators

Coefficient Quantization

- **Bit Width**: User-definable from 1 to 32 bits
- **Signed/Unsigned**: Support for both signed and unsigned representations
- **Automatic Scaling**: Coefficients normalized to use full bit range
- **Display Formats**: Decimal, Hexadecimal, Binary, Verilog
- **Error Analysis**: MSE, SNR, and maximum error calculations

File Operations

- **Save/Load Configurations**: JSON format with all parameters
- **Export Formats**:
  - C/C++ header files (.h)
  - MATLAB files (.m)
  - Python files (.py)
  - Verilog files (.v)
  - CSV files (.csv)

File Structure

```
fir_filter_generator/
├── main.py          # Main application entry point
├── gui.py           # GUI implementation with tabs
├── math_utils.py    # Filter design and quantization mathematics
├── file_utils.py    # File I/O operations
├── requirements.txt # Required Python packages
└── README.md        # This documentation
```

Installation

1. **Clone or download the project files**
2. **Install required packages:**

   ```bash
   pip install -r requirements.txt
   ```

3. **Run the application:**

   ```bash
   python main.py
   ```

Usage

Filter Design Tab

1. **Select Filter Type**: Choose from dropdown menu
2. **Set Parameters**:
   - Number of coefficients (odd numbers recommended)
   - Cutoff frequencies (normalized: 0.001 to 0.499)
   - Sampling frequency in Hz
3. **View Results**:
   - Real-time frequency response plot
   - Filter coefficients display
   - Cutoff frequency indicators

Quantization Tab

1. **Enable Quantization**: Check the enable checkbox
2. **Configure Quantization**:
   - Set bit width (1-32 bits)
   - Choose signed or unsigned
   - Select display format
3. **View Results**:
   - Quantization error statistics
   - Integer coefficient values
   - Multiple display formats

File Operations

- **Save Config**: Save current filter parameters and coefficients to JSON
- **Load Config**: Load previously saved configurations
- **Export**: Export coefficients in various programming language formats

Mathematical Details

Filter Design

- Uses scipy.signal.firwin() with Hamming window
- Automatic parameter validation and sanitization
- Support for all standard FIR filter types

Quantization Algorithm

1. **Scaling**: Coefficients scaled to use full bit range
2. **Rounding**: Nearest integer quantization
3. **Clipping**: Values clipped to valid bit range
4. **Two's Complement**: Proper handling of signed negative values

Error Metrics

- **MSE**: Mean Squared Error between original and quantized
- **SNR**: Signal-to-Noise Ratio in dB
- **Max Error**: Maximum absolute error

Export Formats

C/C++ Header (.h)

```c
const int16_t filter_coeffs[51] = {
    -123,   -45,    67, ...
};
```

MATLAB (.m)

```matlab
filter_coeffs = [
    -0.0012345,  0.0023456, ...
];
```

Verilog (.v)

```verilog
reg signed [15:0] filter_coeffs [0:50];
initial begin
    filter_coeffs[0] = 16'hFF85;
    ...
end
```

Technical Requirements

- Python 3.6+
- NumPy 1.19.0+
- SciPy 1.5.0+
- Matplotlib 3.3.0+
- Tkinter (usually included with Python)

License