ECG SIGNAL PROCESSING

Workshop Guide

A Practical Introduction to Biomedical Signal Processing in MATLAB

Dr Mahdi Torabi
Torabi Signals Ltd.
torabisignals.com

About This Workshop

This workshop takes you step by step through the fundamentals of ECG (Electrocardiogram) signal processing using MATLAB. No prior signal processing experience is needed. Each section builds on the previous one, and every step has clear tasks for you to try.

Signal used: Synthetic single-channel ECG | 256 Hz | 256 seconds | 65,536 samples
MATLAB file: ECG_Workshop.m (run section by section using Ctrl+Enter)

What is an ECG and Why Does It Matter?

An Electrocardiogram (ECG) is a recording of the electrical activity of the heart over time. Every time the heart beats, it produces a small electrical signal that travels through the body and can be picked up by electrodes placed on the skin. This recording is plotted as a waveform against time and is one of the most important diagnostic tools in modern medicine.

The PQRST Waveform

Each heartbeat in an ECG produces a recognisable shape called the PQRST complex. Each letter refers to a different part of the heart's electrical cycle:

Wave	What it represents
P wave	The upper chambers (atria) contracting to push blood into the ventricles
QRS complex	The main pumping action — ventricles contracting to push blood to the body
T wave	The ventricles recovering and preparing for the next beat

The tall sharp spike in the middle — called the R-peak — is the most visible feature. By counting R-peaks over time, we can calculate heart rate in beats per minute (BPM).

Why ECG Signal Processing Matters

In the real world, ECG signals are never perfectly clean. They are corrupted by various types of noise from the patient's movement, breathing, electrical equipment nearby, and electrode contact issues. Before any clinical analysis can be performed — such as detecting arrhythmias or measuring heart rate — the signal must be cleaned and processed.

Why this matters in the medical industry

- ECG devices in hospitals must filter powerline noise (50/60 Hz) in real time
- Wearable heart monitors use signal processing to detect abnormal rhythms
- Automated ECG analysis software relies on clean signals to diagnose correctly
- Poor signal quality leads to misdiagnosis — signal processing is a patient safety issue
- Ambulatory monitors record for 24–48 hours, generating huge amounts of noisy data that must be processed automatically

© 2026 Torabi Signals Ltd. | torabisignals.com

Workshop At a Glance

The workshop is divided into 6 sections. Each section is one block of MATLAB code that you run using Ctrl+Enter. Below is a quick summary of what each section covers.

Section	Title	What you will do
1	**Load ECG Data**	Load the ECG file, set the sampling rate to 256 Hz, and confirm the signal length and amplitude range
2	**Plot Full ECG**	Display the entire 256-second recording and observe the repeating heartbeat pattern
3	**Zoom into 5 Seconds**	Zoom into a short window to clearly see individual P, QRS, and T waves
4	**Add Noise**	Corrupt the clean ECG with two real-world noise types: White Gaussian Noise and 50 Hz Powerline Interference
5	**Filter the Noise**	Apply a Bandpass filter to remove WGN and a Notch filter to remove 50 Hz powerline noise
6	**FFT Analysis**	Use the Fast Fourier Transform to reveal the heartbeat frequency and visually identify the 50 Hz noise spike

© 2026 Torabi Signals Ltd. | torabisignals.com

Section 1 Load ECG Data

Before any processing can begin, we need to load the ECG data into MATLAB and understand what we are working with. This first step is the foundation of every signal processing workflow.

What the code does

- Loads ecg_data.mat from your current MATLAB folder
- Sets the sampling frequency to 256 Hz — this tells MATLAB how many samples were recorded per second
- Builds a time axis in seconds, so we can plot the signal against real time rather than sample numbers
- Prints a summary to the Command Window showing duration, number of samples, and amplitude range

Key concept: Sampling Frequency

> The sampling frequency (fs = 256 Hz) means the ECG was recorded 256 times per second. This is important because it determines the highest frequency we can detect in the signal (called the Nyquist frequency, which is fs/2 = 128 Hz). For ECG signals, 256 Hz is more than enough — the clinically useful content sits below 40 Hz.

Why this matters

Every signal processing task starts with knowing the sampling rate. Without it, you cannot calculate correct frequency axes for the FFT, set filter cutoffs in Hz, or interpret the time axis correctly. Getting this right from the start avoids errors in every step that follows.

Workshop task — try this in MATLAB:

1. After running Section 1, type whos in the Command Window. What variables do you see?
2. Can you calculate the recording duration manually? (samples ÷ fs)
3. What is the amplitude range of the signal? Does it look like a typical ECG (millivolts)?

© 2026 Torabi Signals Ltd. | torabisignals.com

Section 2 Plot the Full ECG Signal

The first thing any engineer or clinician does with a new ECG is look at it. Plotting the whole recording gives you a global view — you can see the rhythm, spot obvious artefacts, and get a feel for the signal quality before doing any detailed analysis.

What the code does

- Plots the entire 256-second ECG recording on a single figure
- Uses a real time axis in seconds on the x-axis (not sample numbers)
- Labels axes clearly with units — time in seconds, amplitude in millivolts (mV)

What to look for

You should see a regular repeating pattern of tall spikes — each one is one heartbeat. The pattern should look consistent and clean. At this zoom level, individual P, Q, R, S, T waves are too small to see clearly, but you can estimate the heart rate by counting the spikes.

> **Quick heart rate estimate from the full plot:**
> Count the number of tall spikes (R-peaks) visible in any 10-second portion, then multiply by 6 to get an approximate BPM. A normal resting heart rate is 60–100 BPM.

Why this matters

Visualisation is the most immediate quality check in biomedical signal processing. Before running algorithms, an engineer always looks at the raw signal to check for obvious problems such as flat lines (lost electrode contact), sudden jumps (motion artefact), or signal that is far too noisy to process reliably.

> **Workshop task — try this in MATLAB:**
> 1. Run Section 2 and observe the full recording.
> 2. Count the approximate number of heartbeats visible. Estimate the heart rate in BPM.
> 3. Does the signal look consistent throughout, or are there any irregular sections?

© 2026 Torabi Signals Ltd. | torabisignals.com

Section 3 Zoom Into a 5-Second Window

The full recording is useful for checking the overall rhythm, but individual heartbeat features are too small to see at that scale. This section zooms into a short 5-second window so you can clearly see the shape of each PQRST complex.

What the code does

- Displays two subplots: the full ECG on top with a highlighted yellow region, and a zoomed 5-second view on the bottom
- The yellow highlight shows exactly which part of the full signal you are zoomed into
- You can change the start_sec variable to explore different parts of the recording

How to estimate heart rate manually

> Count the number of R-peaks (the tall sharp spikes) in the 5-second window, then multiply that count by 12. This gives you the heart rate in beats per minute (BPM). For example, if you count 6 peaks in 5 seconds: 6 × 12 = 72 BPM — a perfectly normal resting heart rate.

Why this matters

In clinical practice, cardiologists look at short ECG windows (often 10 seconds on a standard 12-lead ECG printout) to identify arrhythmias, measure intervals, and assess waveform morphology. The ability to zoom in and navigate through a recording is a core part of ECG analysis software.

Workshop task — try this in MATLAB:

1. Run Section 3 and count the R-peaks in the 5-second window. Calculate the BPM.
2. Change start_sec to 60, then to 120. Is the heart rate consistent throughout the recording?
3. Can you clearly identify the P wave, QRS complex, and T wave in the zoomed view?

© 2026 Torabi Signals Ltd. | torabisignals.com

Section 4 Add Noise to the ECG

Real-world ECG recordings are never as clean as our synthetic signal. In this section, we deliberately corrupt the signal with two types of noise that are commonly encountered in clinical environments. Understanding how noise looks and behaves is the first step to removing it.

Noise Type 1: White Gaussian Noise (WGN)

- Appears as random high-frequency fuzz across the entire signal
- Caused by: electronic amplifier noise, electrode contact resistance, and muscle artefacts (EMG)
- Spread equally across all frequencies — there is no single frequency you can target
- Controlled by SNR (Signal-to-Noise Ratio) in decibels (dB) — higher dB means less noise

Noise Type 2: 50 Hz Powerline Interference

- Appears as a fast, regular ripple sitting on top of the ECG waveform
- Caused by: electromagnetic coupling from mains electricity (50 Hz in UK/Europe, 60 Hz in USA)
- A pure sine wave at exactly one frequency — which makes it very easy to remove
- One of the most common artefacts in clinical ECG recordings worldwide

Understanding SNR

SNR value	What it means for the ECG
30 dB	Very little noise — signal is mostly clean, noise is barely visible
20 dB	Moderate noise — ECG shape still clear but some fuzz visible (default)
10 dB	Heavy noise — ECG shape partially obscured
5 dB	Severe noise — ECG features hard to identify

Why this matters

Every ECG device on the market — from a hospital bedside monitor to a smartwatch — has to deal with these exact noise types. Understanding what they look like is essential before you can design the right filter to remove them.

Workshop task — try this in MATLAB:

1. Run Section 4 and compare the two noisy signals side by side with the clean original.
2. Change snr_wgn to 5 dB. Can you still identify the heartbeat shape?
3. Change powerline_amp to 0.6 mV. How does the signal look compared to the original?
4. Which type of noise is easier to identify just by looking at the time-domain plot?

© 2026 Torabi Signals Ltd. | torabisignals.com

Section 5 Pre-Processing: Filter the Noise

Now that we understand what the noise looks like, we apply filters to remove it. Different noise types require different filter strategies. This section introduces two of the most important filters used in clinical ECG processing.

Filter 1: Bandpass Filter — removes White Gaussian Noise

A bandpass filter keeps only the frequencies between two cutoff values and removes everything else.

- Lower cutoff: 0.5 Hz — removes any very slow drift
- Upper cutoff: 40 Hz — removes high-frequency noise (WGN lives above this)
- The ECG's clinically useful content sits entirely within 0.5–40 Hz
- Filter type: Butterworth — smooth response with no ripple in the passband
- filtfilt() is used for zero-phase filtering — the output stays perfectly aligned in time

Filter 2: Notch Filter — removes 50 Hz Powerline Noise

A notch filter removes a very narrow band of frequencies around a single target, leaving everything else completely untouched.

- Target frequency: 50 Hz (exactly where the powerline noise sits)
- Bandwidth: ±1 Hz — only a tiny window is removed, nothing else is affected
- This is the standard approach used in all clinical ECG devices
- The result looks almost identical to the original — the ripple simply disappears

What the figure shows

The Section 5 figure has 5 panels arranged in a clear layout:

- Top (full width): Original clean ECG as a reference
- Middle left: ECG corrupted by WGN
- Middle right: The same signal after the Bandpass filter — compare to the reference
- Bottom left: ECG corrupted by 50 Hz powerline noise
- Bottom right: The same signal after the Notch filter — the ripple is gone

Why this matters

Signal filtering is the single most important pre-processing step in biomedical signal processing. Automated analysis algorithms — such as R-peak detectors or arrhythmia classifiers — will fail or produce errors if applied to noisy data. Filtering comes first, always.

Workshop task — try this in MATLAB:

1. Run Section 5 and compare each noisy signal with its filtered version.
2. Change bp_high to 20 Hz. Does the ECG waveform shape change? Why?
3. Change notch_bw to 4 Hz. Does the filtered result look different?
4. Compare the filtered signals to the original (dashed grey line). How closely does each filter recover the original?

© 2026 Torabi Signals Ltd. | torabisignals.com

Section 6 Frequency Analysis — FFT

The Fast Fourier Transform (FFT) is one of the most powerful tools in signal processing. Everything we have done so far has been in the time domain — we looked at the signal over time. The FFT moves us to the frequency domain, where we can see exactly which frequencies are present in a signal and how much energy each one contains.

Time domain vs. frequency domain

Domain	What you see
Time domain	The signal's amplitude changing over time — the familiar ECG waveform
Frequency domain (FFT)	Which frequencies are present and how strong each one is

What the FFT reveals in this workshop

Heartbeat frequency

The FFT of the clean ECG shows a peak at a low frequency (typically 1–2 Hz). This corresponds directly to the heart rate. If the peak is at 1.2 Hz, the heart rate is 1.2 × 60 = 72 BPM. This is how the code automatically calculates and displays the BPM from the signal.

50 Hz powerline spike

The FFT of the powerline-corrupted ECG shows a sharp, unmistakable spike at exactly 50 Hz. In the time domain, this noise just looks like a fuzzy ripple — it is hard to characterise. In the frequency domain, it is instantly obvious. This is one of the key advantages of FFT analysis.

What the figure shows

- Top panel (full width): Both FFTs overlaid — clean ECG in blue, powerline-corrupted ECG in purple. The 50 Hz spike clearly stands out
- Bottom left: Clean ECG FFT with the heartbeat peak automatically identified and labelled with its frequency and equivalent BPM
- Bottom right: Noisy ECG FFT with the 50 Hz spike clearly marked — this is the noise fingerprint

Why this matters

In every ECG device and biosignal processing system, the FFT is used to identify noise sources, verify filter performance, and detect periodic signals hidden within the data. The 50 Hz notch filter used in Section 5 was only possible because the FFT showed us exactly where the noise was. This connection between frequency analysis and filter design is fundamental to the entire field of biomedical signal processing.

Workshop task — try this in MATLAB:

1. Run Section 6 and look at the top overlaid FFT plot. Can you immediately spot the 50 Hz spike?
2. Read the heartbeat frequency from the bottom-left plot. Multiply by 60 — does it match your manual count from Section 3?

© 2026 Torabi Signals Ltd. | torabisignals.com

3. Go back to Section 4 and change powerline_amp to 0.1 mV. Re-run Sections 4 and 6. Is the 50 Hz spike still visible in the FFT?
4. Can you see why the FFT is so useful for noise identification compared to just looking at the time-domain signal?

Quick Reference

Key Parameters You Can Change

Parameter	Default	What it controls
start_sec (Section 3)	10	Start time of the 5-second zoom window
snr_wgn (Section 4)	20 dB	Strength of the White Gaussian Noise (lower = noisier)
powerline_amp (Section 4)	0.3 mV	Amplitude of the 50 Hz powerline sine wave
bp_low (Section 5)	0.5 Hz	Lower cutoff of the bandpass filter
bp_high (Section 5)	40 Hz	Upper cutoff of the bandpass filter
notch_bw (Section 5)	2 Hz	Width of the 50 Hz notch (frequency band removed)

Glossary of Key Terms

Term	Meaning
ECG / EKG	Electrocardiogram — a recording of the heart's electrical activity over time
PQRST complex	The characteristic shape of one heartbeat in an ECG signal
R-peak	The tallest spike in the QRS complex — used to detect each heartbeat
Sampling frequency (fs)	Number of samples recorded per second (256 Hz in this workshop)
SNR	Signal-to-Noise Ratio — higher dB means less noise
White Gaussian Noise	Random noise spread evenly across all frequencies
Powerline interference	50 Hz (or 60 Hz) noise from mains electricity coupling into ECG leads
Bandpass filter	Keeps a range of frequencies and removes everything outside that range
Notch filter	Removes a very narrow band at one specific frequency, leaving everything else
FFT	Fast Fourier Transform — converts a signal from time domain to frequency domain
BPM	Beats Per Minute — the heart rate, calculated as heartbeat frequency × 60
Nyquist frequency	The maximum detectable frequency = fs / 2 = 128 Hz for this signal

About This Workshop

This workshop was developed by Dr Mahdi Torabi at Torabi Signals Ltd. as a practical introduction to biomedical signal processing for students, engineers, and healthcare professionals. The MATLAB code and supporting materials are designed to be accessible to anyone with a basic understanding of MATLAB, regardless of prior signal processing experience.

Contact and Resources

Website: https://torabisignals.com
Email: info@torabisignals.com
License: Educational and personal use. See license header in the MATLAB file.

© 2026 Torabi Signals Ltd. All rights reserved. This document and the accompanying MATLAB code are the intellectual property of Dr Mahdi Torabi and Torabi Signals Ltd.