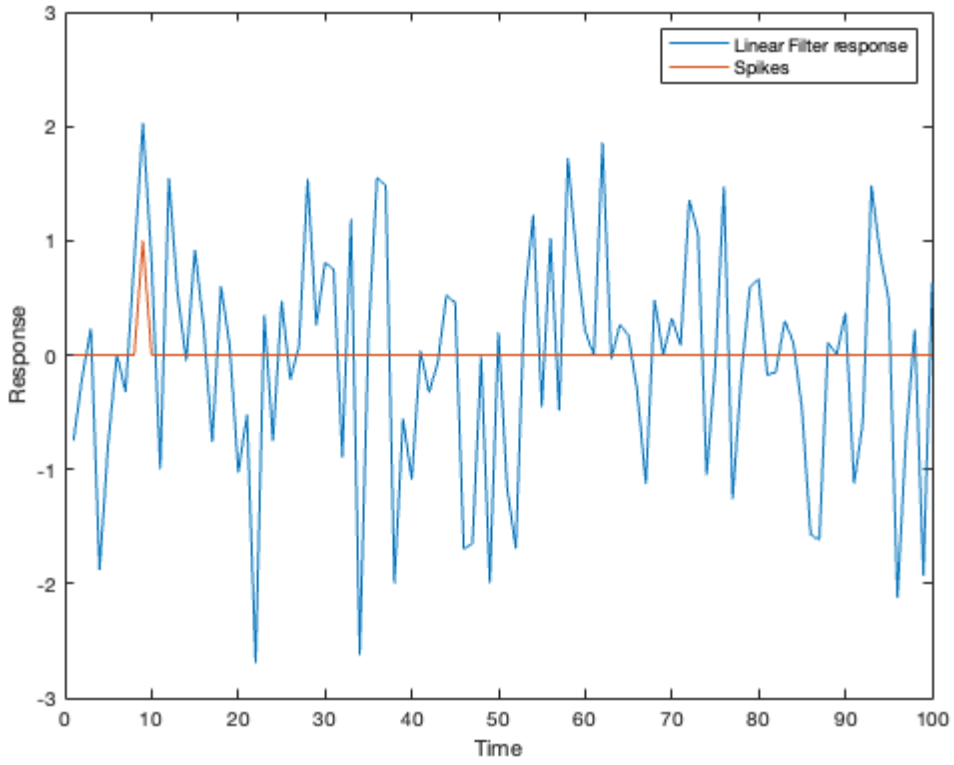
Contents


```
clear; close all; clc;
```

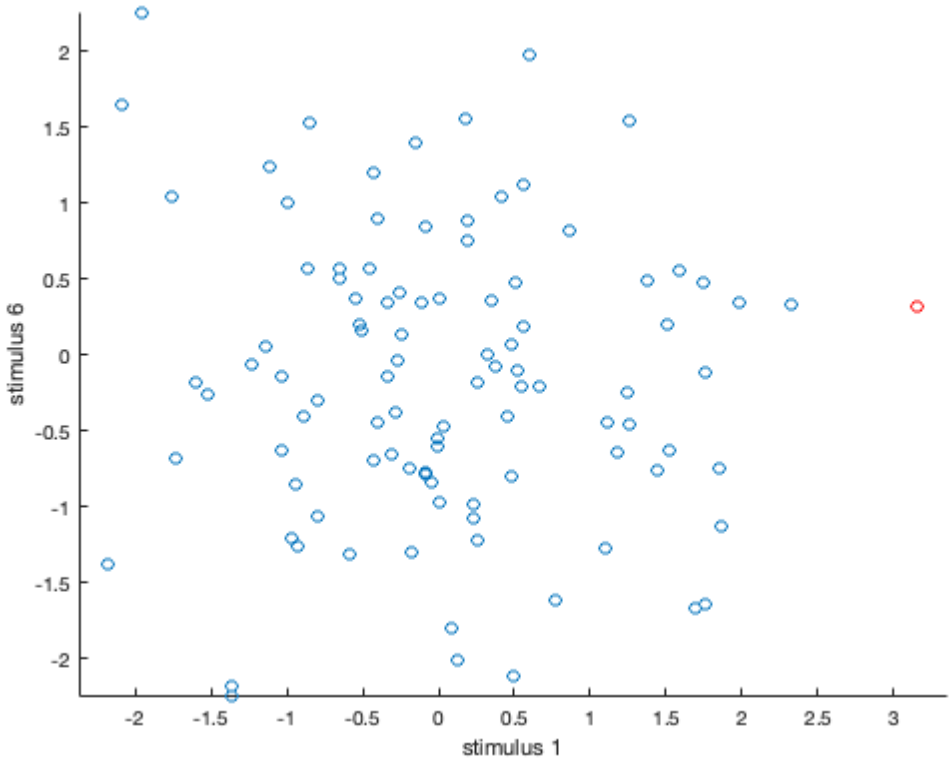
a)

```
kernel_squared = [1 2 1; 2 4 2; 1 2 1]/6;
kernel = kernel_squared(:);
duration = 100;
[spikes, stimuli] = runGaussNoiseExpt(kernel, duration);

lin_filter_response = stimuli * kernel;
figure()
plot(lin_filter_response, 'DisplayName', 'Linear Filter response')
hold on;
plot(spikes, 'DisplayName', 'Spikes');
legend()
xlabel('Time')
ylabel('Response')
```



```
figure();
scatter(stimuli(:, 1), stimuli(:, 6));
hold on;
scatter(stimuli(spikes, 1), stimuli(spikes, 6), 'r')
xlabel('stimulus 1')
ylabel('stimulus 6')
axis equal;
```



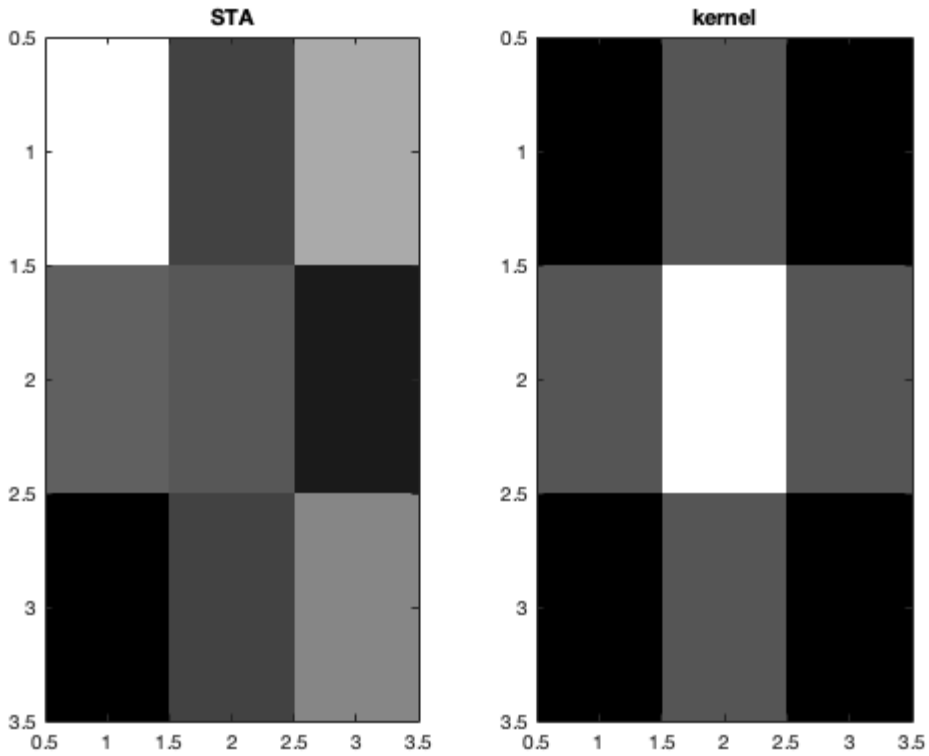
b)

```
scaled_stimuli = stimuli - mean(stimuli);
spike_count = sum(spikes);
STA = (scaled_stimuli' * spikes)./spike_count;
STA_normalized = STA./(sqrt(sum(STA.^2)));

STA_squared = reshape(STA_normalized, [3, 3]);

figure()
subplot(1, 2, 1)
imagesc(STA_squared)
colormap(gray)
title('STA')

subplot(1, 2, 2)
imagesc(kernel_squared)
colormap(gray)
title('kernel')
```



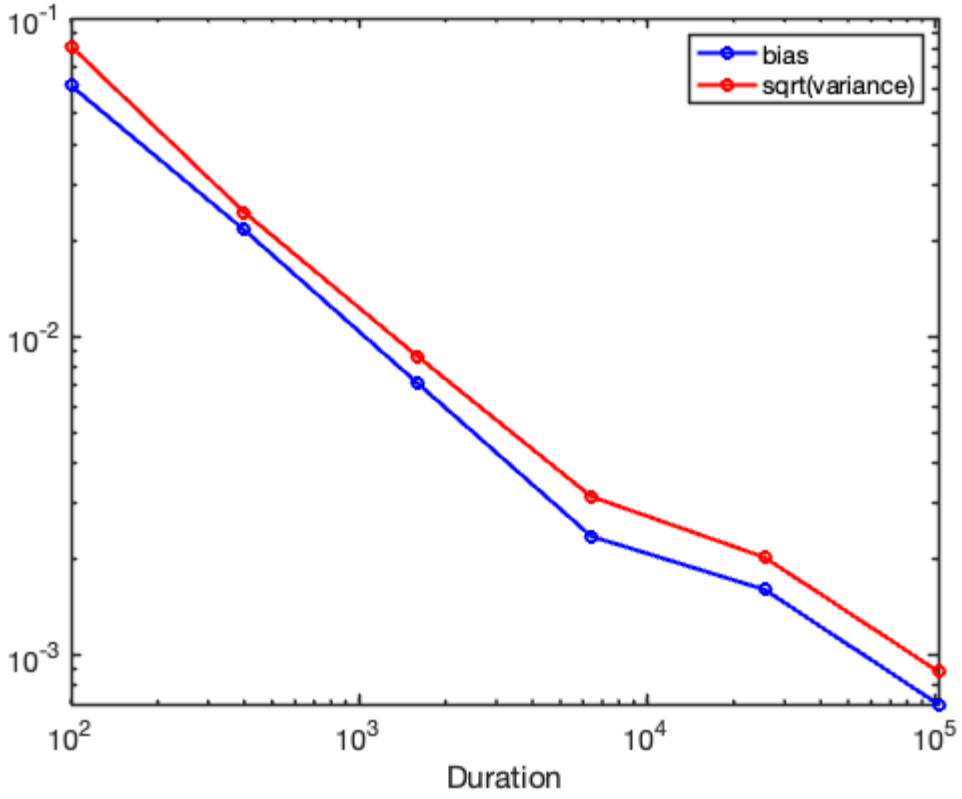

```
durations = [100, 400, 1600, 6400, 25600, 102400];
runs = 100;

STA_across_runs = zeros(length(durations), runs, length(kernel));
for i = 1:length(durations)
    duration = durations(i);
    for run = 1:runs
        STA_across_runs(i, run, :) = get_STA(kernel, duration);
    end
end

mean_STA_across_runs = squeeze(mean(STA_across_runs, 2));

kernel_oned = ones(length(durations), 1) * kernel';
diff_kernel_STA = kernel_oned - mean_STA_across_runs;
ave_diff_kernel = mean(abs(diff_kernel_STA), 2); % estimation bias
squared_diff_kernel_STA = (kernel_oned - mean_STA_across_runs).^2;
var_diff_kernel = mean(squared_diff_kernel_STA, 2);

figure()
h = loglog(durations, ave_diff_kernel, 'bo-', ...
    durations, sqrt(var_diff_kernel), 'ro-');
xlabel('Duration')
set(h(1), 'LineWidth', 2)
set(h(2), 'LineWidth', 2)
set(gca, 'LineWidth', 1.5)
set(gca, 'FontSize', 14)
legend('bias', 'sqrt(variance)')
```



c)

```
duration = 6400;
[spikes, stimuli] = runGaussNoiseExpt(kernel, duration);
scaled_stimuli = stimuli - mean(stimuli);
spike_count = sum(spikes);
if spike_count <= 0.01
    STA_normalized = 0;
else
    STA = (scaled_stimuli' * spikes)./(spike_count);
    STA_normalized = STA./(sqrt(sum(STA.^2)));
end
```

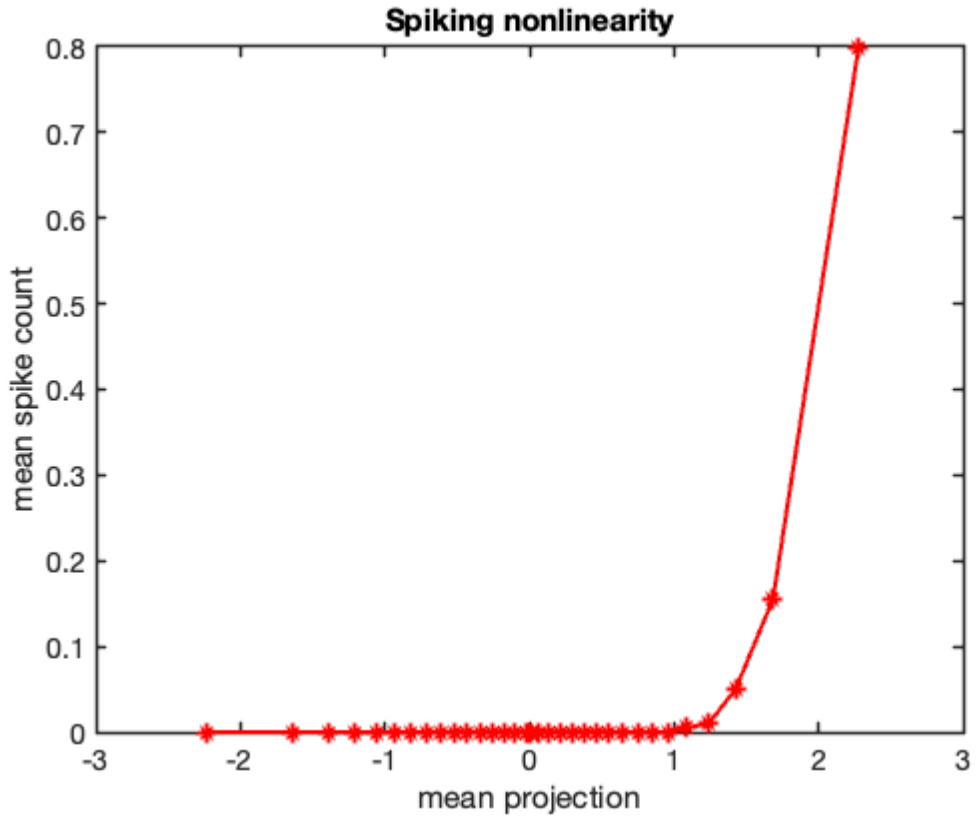
```
stimuli_projection = stimuli * STA_normalized;
[stimuli_projection_sorted, proj_index] = sort(stimuli_projection, 'descend');
spike_reordered = spikes(proj_index);

bin_width = 200;
bin_size = 1:bin_width:duration;
stimuli_projection_mean = zeros(length(bin_size));
spike_reordered_mean = zeros(length(bin_size));

for bin_ = 1:length(bin_size)
    bin_lb = bin_size(bin_);
    bin_ub = bin_width * bin_;

    stimuli_projection_mean(bin_) = mean(stimuli_projection_sorted(bin_lb: bin_ub), 1);
    spike_reordered_mean(bin_) = mean(spike_reordered(bin_lb: bin_ub), 1);
end
```

```
figure()
plot(stimuli_projection_mean, spike_reordered_mean, 'r*-', 'LineWidth', 2)
set(gca, 'LineWidth', 1.5)
set(gca, 'FontSize', 14)
xlabel('mean projection')
ylabel('mean spike count')
title('Spiking nonlinearity')
```



d)

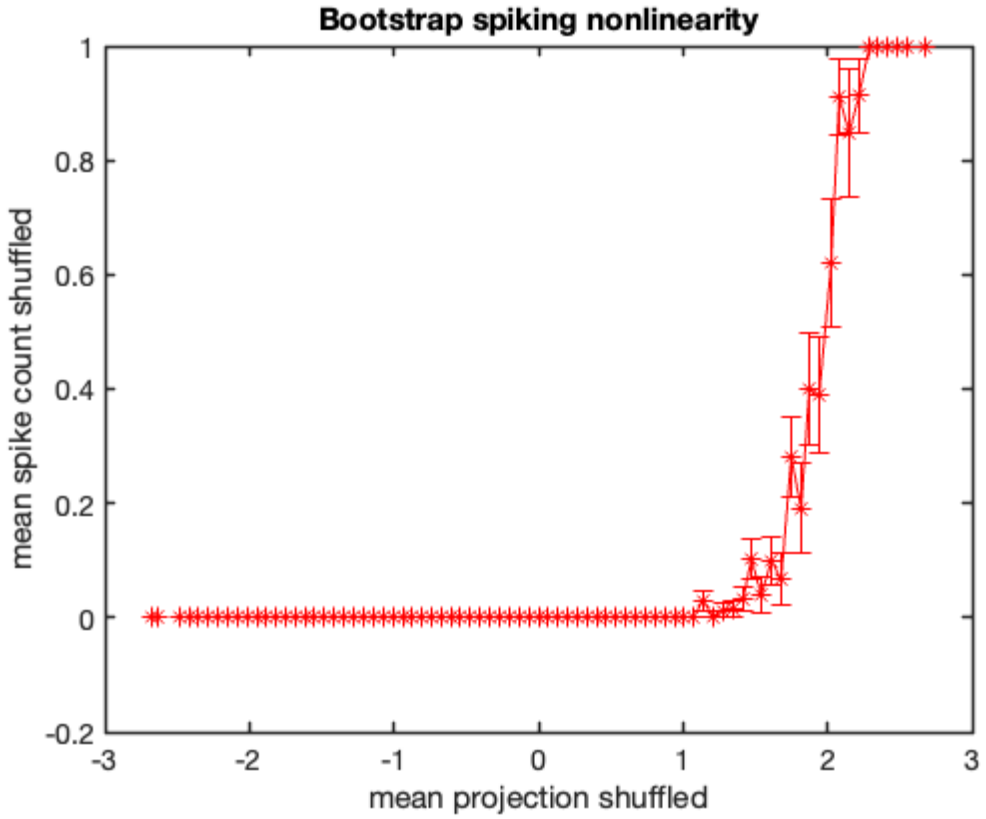
```
bootstrap_times = 100;
numBins = 120;

stimuli_proj_min = floor(min(stimuli_projection_sorted));
stimuli_proj_max = ceil(max(stimuli_projection_sorted));
bins = linspace(stimuli_proj_min, stimuli_proj_max, numBins);
stimuli_projection_shuffled_mean = zeros(numBins, bootstrap_times);
spike_reordered_shuffled_mean = zeros(numBins, bootstrap_times);

for i = 1:bootstrap_times
    new_indices = randi(duration, 1, duration);
    stimuli_projection_shuffled = stimuli_projection(new_indices);
    spike_reordered_shuffled = spikes(new_indices);
    dicret_binned_idx = discretize(stimuli_projection_shuffled, bins);
    for kk = 1:numBins
        stimuli_projection_shuffled_binned = stimuli_projection_shuffled(dicret_binned_idx == kk);
        stimuli_projection_shuffled_mean(kk, i) = mean(stimuli_projection_shuffled_binned, 1);
        spike_reordered_shuffled_binned = spike_reordered_shuffled(dicret_binned_idx == kk);
        spike_reordered_shuffled_mean(kk, i) = mean(spike_reordered_shuffled_binned, 1);
    end
end
```

```
stimuli_projection_shuffled_bootstrap_mean = mean(stimuli_projection_shuffled_mean, 2);
stimuli_projection_shuffled_bootstrap_stdev = std(stimuli_projection_shuffled_mean, 0, 2);
spike_reordered_shuffled_bootstrap_mean = mean(spike_reordered_shuffled_mean, 2);
spike_reordered_shuffled_bootstrap_stdev = std(spike_reordered_shuffled_mean, 0, 2);

figure()
errorbar(stimuli_projection_shuffled_bootstrap_mean, ...
    spike_reordered_shuffled_bootstrap_mean, ...
    spike_reordered_shuffled_bootstrap_stdev, 'r*-', 'LineWidth', 1)
set(gca, 'LineWidth', 1.5)
set(gca, 'FontSize', 14)
xlabel('mean projection shuffled')
ylabel('mean spike count shuffled')
title('Bootstrap spiking nonlinearity')
```




```
function STA_normalized = get_STA(kernel, duration)
    [spikes, stimuli] = runGaussNoiseExpt(kernel, duration);
    scaled_stimuli = stimuli - mean(stimuli);
    spike_count = sum(spikes);
    if spike_count <= 0.01
        STA_normalized = 0;
    else
        STA = (scaled_stimuli' * spikes)./(spike_count);
        STA_normalized = STA./(sqrt(sum(STA.^2)));
    end
end
```

Published with MATLAB® R2021b