Warsaw University of Technology

Faculty of Electronics and Information Technology

ENUME 2025 – Assignment B #05

Approximation of Functions

Tutor: DSc. Kajetana Marta Snopek

Authors: Qinghe Zhao, Yi Yang

Warsaw, 26.04.2025

Table of Contents

List of Mathematical Symbols

$x(t)$ – original periodic signal:

$$x(t) = \begin{cases} Acos(\omega_0 t), t \in \left(-\frac{T}{4}, \frac{T}{4}\right] \\ 0, \qquad\qquad t \in \left(\frac{T}{4}, \frac{3T}{4}\right] \end{cases}$$

f_0 – signal frequency.

ω_0 – signal angular frequency, defined as:

$$\omega_0 = 2\pi f_0$$

T – period of the signal, defined as:

$$T = \frac{1}{f_0}$$

A – signal amplitude.

t – continuous time variable.

t_i – discrete time vector, defined as:

$$t_i = 0: h: T - h$$

H_{size} – Hadamard matrix size, defined as:

$$H_{size} = 2^{log_2(M_{\max})}$$

H – the Hadamard matrix used to build Walsh function, defined as:

$$H \in \{\pm 1\}^{H_{size} \times H_{size}}$$

idx – the index of the sub-interval (column) in H, defined as:

$$idx = \left\lfloor \frac{(t \bmod T)}{T} \; H_{size} \right\rfloor + 1$$

$\phi_n(t)$ – the n-th Walsh function based on [0,T], defined as:

$$\phi_n(t) = H(n+1, idx)$$

n – the index of the Walsh function, which is the row index in the matrix H.

N – number of sub-intervals for the rectangle method.

N_{vect} – list of N values:

$$N_{vect} = [32, 64, 128, 256, 512, 1024]$$

h – integration step size (the width of each sub-interval), defined as:

$$h = \frac{T}{N}$$

$c_{n_{rect}}$ – coefficient of $\phi_n(t)$ in rectangle method, defined as:

$$c_{n_{rect}}(n+1, k) = \frac{h}{T} \sum_{i=0}^{N-1} x(t_i)_{rect} \phi_n(t_i)$$

$$t_i = i \times h$$

$c_{n_{quad}}$ – coefficient of $\phi_n(t)$ in build-in function, defined as:

$$c_{n_{quad}}(n+1) = \frac{1}{T} \int_0^T x(t) \cdot \phi_n(t) \; dt$$

M – truncation order in Fourier-Walsh series.

M_{list} – list of M values:

$$M_{list} = [3, 5, 8, 16]$$

M_{max} – the maximum value from M_{list}

$x_M(t)\ _R$ – Fourrier-Walsh Series using rectangle method, defined as:

$$x_M(t)\ _R = \sum_{n=0}^{M-1} c_{n_{rect}} \cdot \phi_n(t)$$

$x_M(t)\ _Q$ – Fourrier-Walsh Series using build-in function, defined as:

$$x_M(t)\ _Q = \sum_{n=0}^{M-1} c_{n_{quad}} \cdot \phi_n(t)$$

$N_{\text{t\_plot}}$ – the total number of evenly-spaced time used to form t_{dense}.

t_{dense} – dense time grid over one period.

$WalshMat$ – matrix of Walsh basis sampled on the dense grid.

x_{true} – the signal sampled on the t_{dense} used as reference.

E_0 – the total energy of the signal $x(t)$

ε_M – mean square absolute error, defined as:

$$\varepsilon_M = \int_0^T |x(t) - x_M(t)|^2 \,\mathrm{d}t = E_0 - T \sum_{n=0}^{M-1} |c_n|^2 .$$

Introduction

In this project, we study the approximation of a given periodic signal $x(t)$ using a finite number of terms from its Fourier–Walsh series expansion. The goal is to compute and analyze how well truncated Walsh series approximate the original signal, and how different numerical integration methods affect the accuracy of the series coefficients and the overall reconstruction error.

The original signal $x(t)$ is defined as a piecewise cosine function:

$$x(t) = \begin{cases} Acos(\omega_0 t), t \in \left(-\frac{T}{4}, \frac{T}{4} \right] \\ 0, \qquad\qquad t \in \left(\frac{T}{4}, \frac{3T}{4} \right] \end{cases}$$

The approximation is based on truncating the Fourier–Walsh series to M terms:

$$x_M(t) = \sum_{n=0}^{M-1} c_n \cdot \phi_n(t)$$

The work consists of three parts:

Part 1: We calculate the Fourier–Walsh coefficients cnc_ncn using two methods: (i) the rectangle rule for numerical integration with varying step sizes h, (ii) MATLAB's built-in `integral()` function for high-accuracy quadrature.

Part 2: Using the obtained coefficients, we reconstruct approximations $x_M(t)$ for several truncation orders for $M_{list} = [3, 5, 8, 16, 65]$ and plot them alongside the original signal to visually assess the approximation quality.

Part 3: We calculate and plot the mean-square approximation error ε_M versus $\boldsymbol{M}$ for both methods, and compare how the integration method influences the accuracy of the Fourier–Walsh approximation.

Throughout the project, we also generate Walsh basis functions ordered by sequency to ensure smoother low-order reconstructions, and use a dense sampling grid to guarantee high accuracy in plotting and error evaluation. The conclusions are drawn from the behavior of the Fourier–Walsh reconstructions and the analysis of the mean-square error plot

Methodology and Results of Experiments

□ Generation of Walsh Basis Functions

Walsh functions are an orthonormal set of piecewise-constant ± 1 function on $[0, T]$, which is used in Fourier-Walsh series. To compute the he finite sum $xM(t)$ of M terms of the **Fourier-Walsh series**, we first generate Walsh functions with selected values $M_{list} = [2, 4, 8, 16, 65]$ up to order n = 65, in following steps:

1. Building the raw Hadamard Matrix which provides orthogonal ± 1 patterns. This is the raw structure of Walsh function before imposing a specific ordering. We determine the size as the smallest power of 2 that is larger than M_{max}, because Walsh functions are in a vector space whose dimension must be the power of two. This step ensures that we generate at least M_{max} orthogonal functions in a most efficient way.

2. Compute each row's sequency (the "flips" in each row) and sort rows in ascending order of their sequencies, the Walsh functions are thus arranged from smooth to fast-changing, corresponding to low-to-high frequencies in Fourier series. In this way, we obtain a better low-frequency-like approximation by avoiding fast fliping low-index rows, which introduces unwanted high frequency components in small M reconstructions,

To sort the Hadamard Matrix with row vector:

$$h = [h_1,;\ h_2,;\ \dots,;\ h_N]$$

the sequency for each row is defined as:

$$s = \frac{1}{2}\sum_{j=1}^{N-1} \left| h_{j+1} - h_j \right|$$

This is applied to each row of the Hadamard matrix, then the rows are sorted in ascending order using MATLAB build-in function `sort`. The reordered matrix satisfies:

$$H(1,:) = \mathrm{wal}(0), \quad H(2,:) = \mathrm{wal}(1), \dots$$

Fig.1. presents the first four Walsh functions ordered by sequency.

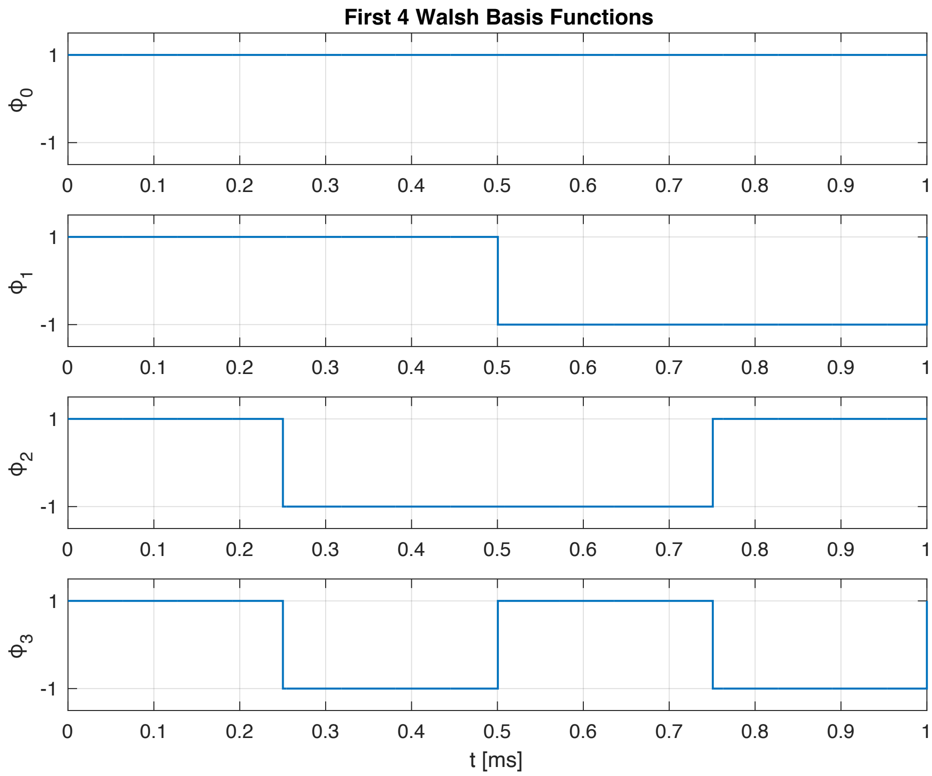

Fig.1. Walsh basis functions ordered by sequency

□ Calculation of the Coefficients of the Fourier-Walsh Series

To approximate the signal $x(t)$ using the Fourier–Walsh series, we must first determine the coefficients c_n for each Walsh basis function $\phi n(t)$, where order $n = 0,1\ldots(M_{max} - 1)$. We compute these coefficients using two methods (i) Rectangle Method and (ii) Build-in Function.

□ Build-in function

This method uses MATLAB's build-in quadrature function `integral()` to compute the Fourier-Walsh coefficients with high precision. For each n, the coefficient is computed by:

$$c_{n_{quad}}(n+1) = \frac{1}{T}\int_0^T x(t) \cdot \phi_n(t)\ dt$$

The results is accurate and serves as the reference solution when evaluating the accuracy of coefficients obtained by the rectangle methods under different h.

 Rectangle Method

To approximate the same integral numerically, we divide the interval $[0, T]$ into N equal sub-intervals:

$$N_{vect} = [32, 64, 128, 256, 512, 1024]$$

Each sub-interval has width h, defined as:

$$h = \frac{T}{N}$$

The coefficient is then approximated using rectangle method:

$$c_{n_{rect}}(n+1, k) = \frac{h}{T} \Sigma_{i=0}^{N-1} x(t_i)_{rect} \phi_n(t_i), \; t_i = i \times h$$

In this way, for each n, we obtain six coefficients $c_{n_{rect}}$ using six different values of h corresponding to the chosen N values. This allows us to analyze how the step size h affects the accuracy of the rectangle approximation.

 Comparison of Accuracy with different step sizes $\boldsymbol{h}$

To investigate the effect of the step size h in the rectangle method, for each combination of n and h, we compare the coefficient $c_{n_{rect}}$ to the reference values $c_{n_{quad}}$ to compute the absolute error:

$$|c_{n_{rect}}(h) - c_{n_{quad}}|$$

The results are shown in Fig.2 for selected indices $n = 3, 7, 11, 15$.

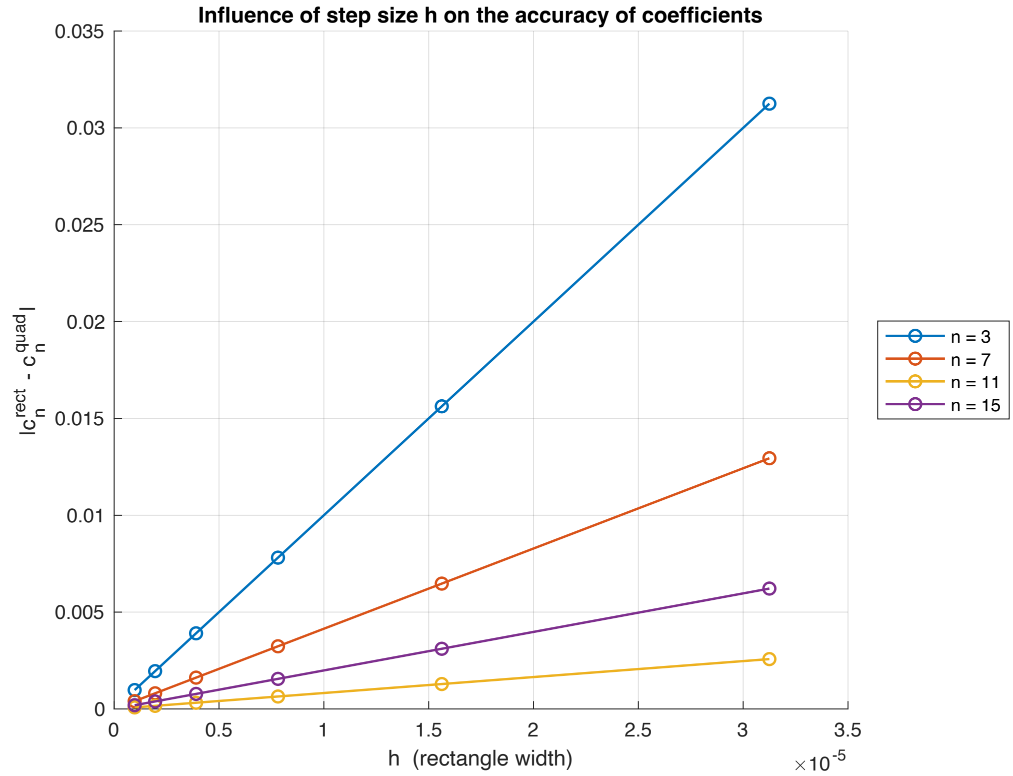

Fig.2. Comparison of accuracy of coefficients obtained by the rectangle methods under different step sizes h.

Next, for signal reconstruction and mean-square error analysis, we select two cases:

the most accurate case with

$$h_1 = \frac{T}{1024} \text{ (“fine h”)}$$

and the least accurate case with

$$h_2 = \frac{T}{32} \text{ (“coarse h”),}$$

in order to compare them with the results obtained by the build-in funciton.

Reconstruction of $x(t)$ and $x_M(t)$ using Fourier-Walsh Series

This step prepares the necessary elements for accurately plotting and comparing the original signal $x(t)$ with its truncated Fourier–Walsh approximations $x_M(t)$, based on coefficients calculated by both integration methods.

Building dense time grid

To accurately reconstruct and evaluate the approximation, we first build a dense time grid over one period:

$$t_{\text{dense}} = [\,0,\,\Delta t,\,2\Delta t,\,\dots,\,T],\quad \Delta t = \frac{T}{N_{\text{t\_plot}}-1},\quad N_{\text{t\_plot}} = 4000$$

t_{dense} contains 4000 evenly-spaced points over $[0,T]$.

Sampling the "ground-truth" signal

Based on t_{dense} we construct the orignal signal $x_{true} = x(t_{dense})$ as a fine-grained reference waveform. We will compare every truncated reconstrunctions $x_M(t)$ with this high-resolution x_{true} to accurately visualize.

Precomputing Walsh basis on the same grid

For efficiency, we precompute the first M_{max} Walsh basis functions on the same grid in to a matrix:

$$\text{WalshMat}(n+1,) = \phi_n(t_{\text{dense}}), n = 0,1,\dots,M_{max}-1.$$

In this way, we can quickly form the partial sums without re-evaluating $\phi_n(t)$ each time when we choose a truncation order M

Truncated Fourier-Walsh Reconstruction

Using the coefficient obtained by rectangle method and build-in function, we reconstruct the approximated signal $x_M(t)\,_R$ and $x_M(t)\,_Q$, the partial sum of the Fourirer -Walsh serie at each order M from $M_{list} = [3,5,8,16]$. This is defined by:

$$x_M(t)_R = \sum_{n=0}^{M-1} c_{n_{rect}} \cdot \phi_n(t)$$

$$x_M(t)_Q = \sum_{n=0}^{M-1} c_{n_{quad}} \cdot \phi_n(t)$$

To improve the early-stage approximation of the signal, we choose odd truncation orders (i.e. $M = 3$ and $M = \mathbf{5}$) for small order M instead of even values (i.e. $M = 2$ and $M = 4$). This is because we observed that at low M, using an even number causes a significantly reduced amplitude in reconstructed signal compared to the true signal, and poorly captured the overall shape.

- **Comparison of different $\boldsymbol{M}$**

For each M, we plot x_{true} together with signals reconstructed using two methods. $x_M(t)_R$ and $x_M(t)_Q$.

Additionally for $x_M(t)_R$, we consider cases with both h_1 (accurate) and h_2 (inaccurate). The results plotted are shown on Fig.3-6.

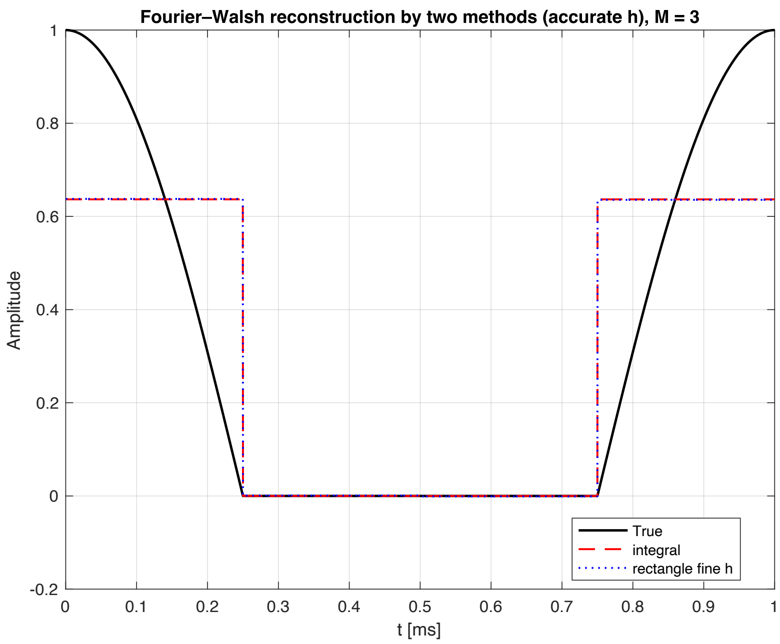

Fig.3. Comparison reconstruction using two methods at M = 3
(with h_1 in rectangle method)

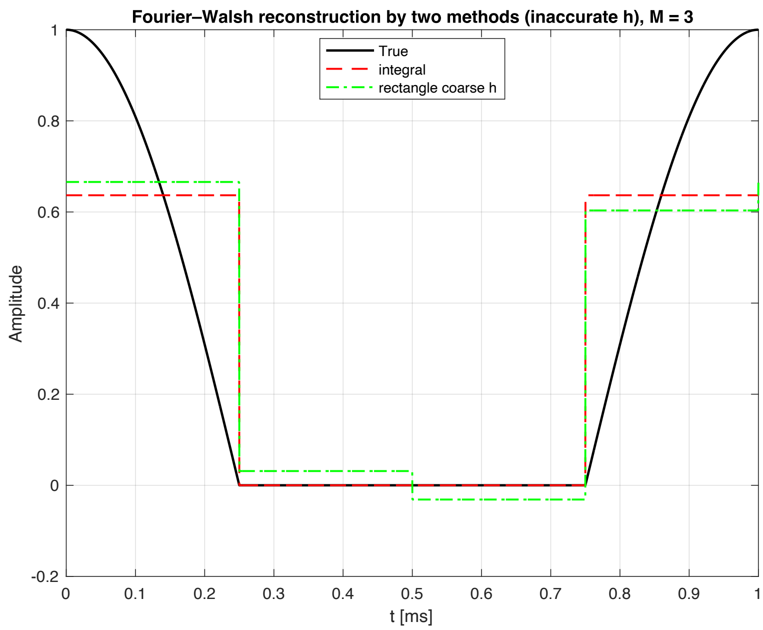

Fig.4. Comparison reconstruction using two methods at M = 3
(with h_2 in rectangle method)

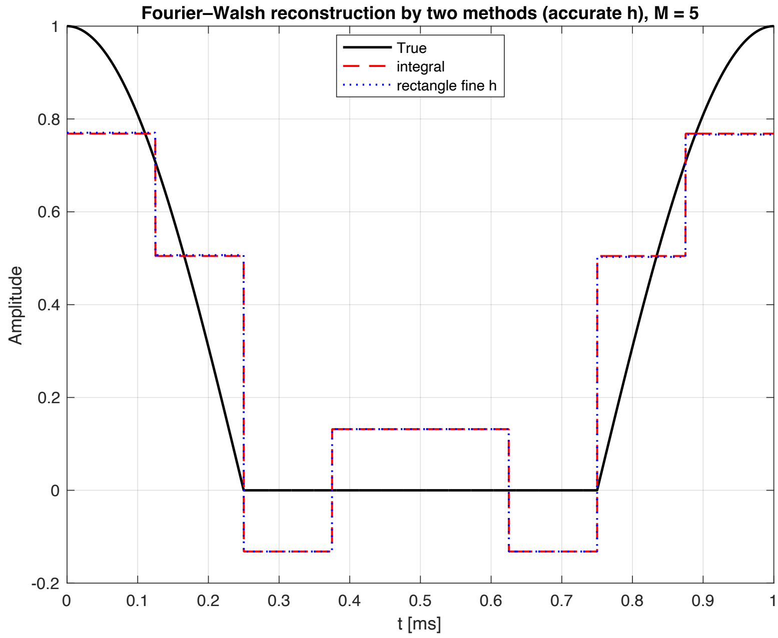

Fig.5. Comparison reconstruction using two methods at M = 5
(with h_1 in rectangle method)

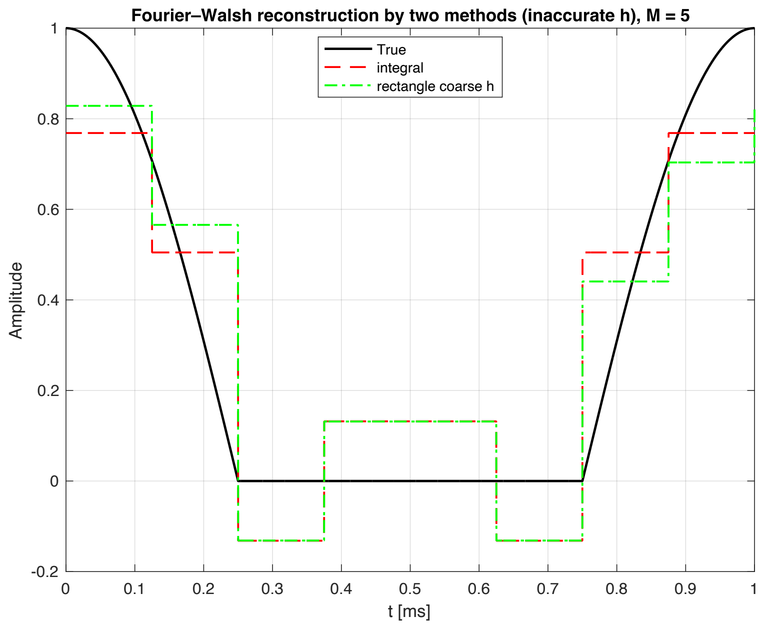

Fig.6. Comparison reconstruction using two methods at M = 5
(with h_2 in rectangle method)

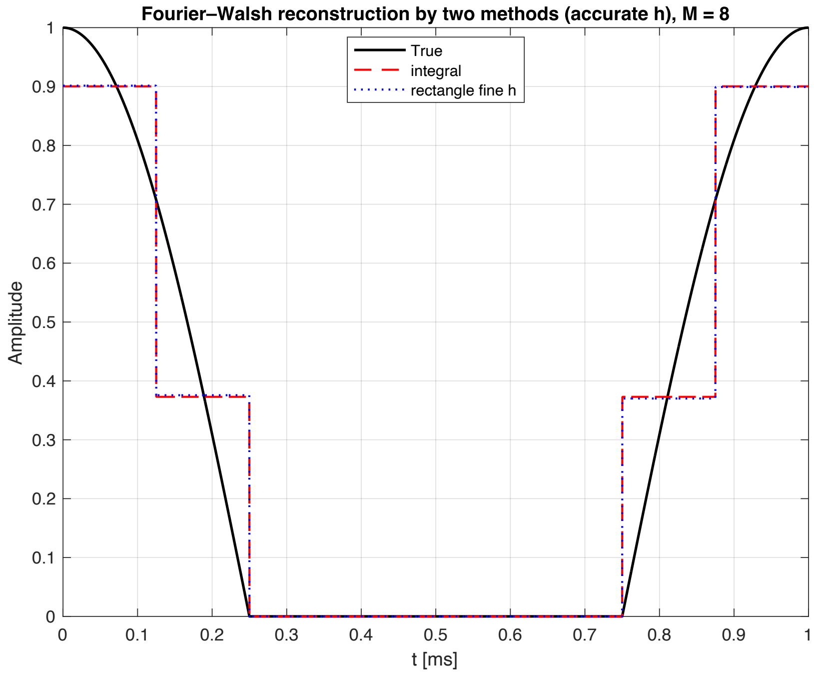

Fig.7. Comparison reconstruction using two methods at M = 8

(with h_1 in rectangle method)

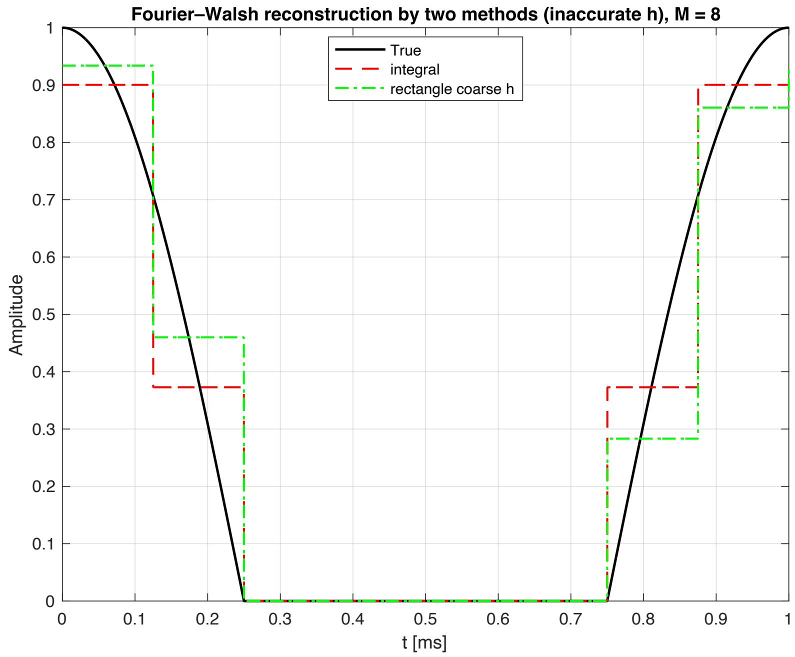

Fig.8. Comparison reconstruction using two methods at M = 8

(with h_2 in rectangle method)

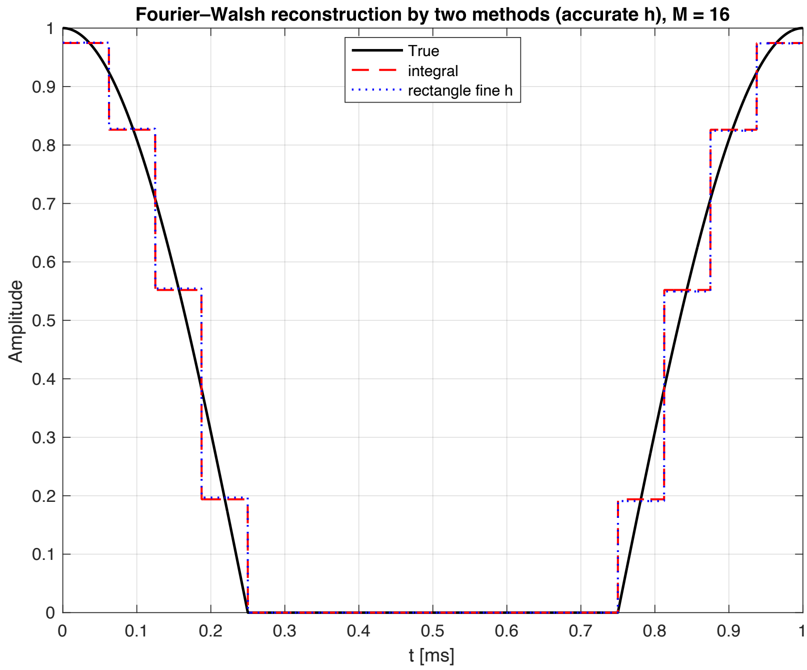

Fig.9. Comparison reconstruction using two methods at M = 16
(with h_1 in rectangle method)

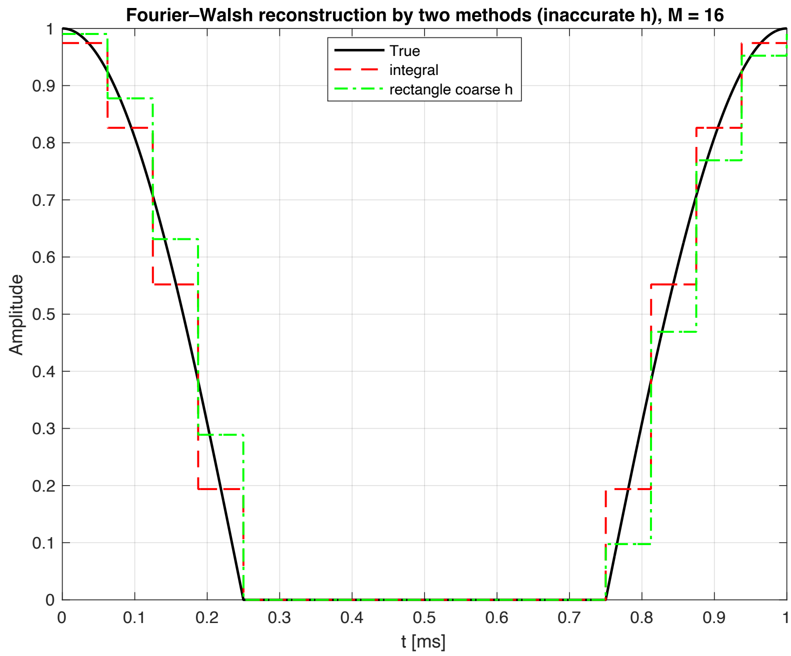

Fig.10. Comparison reconstruction using two methods at M = 16
(with h_2 in rectangle method)

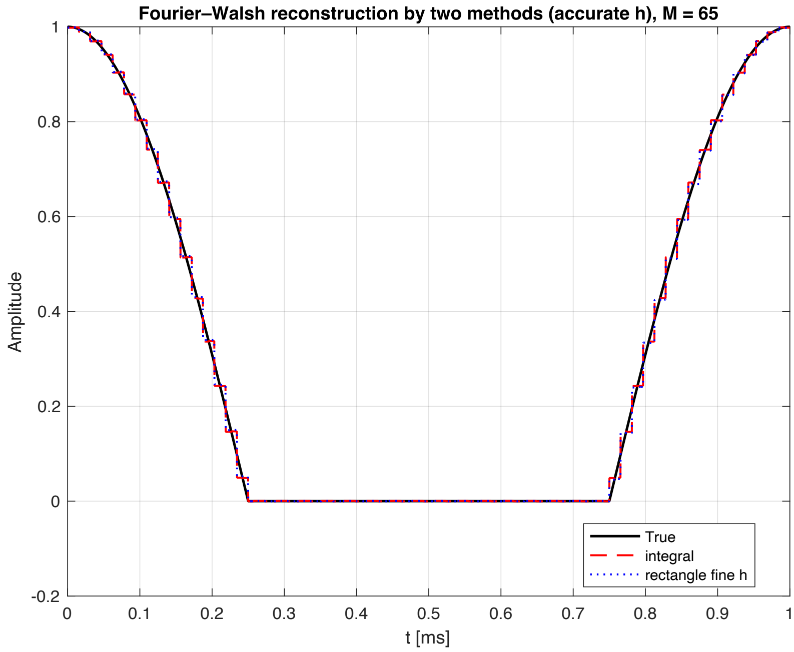

Fig.11. Comparison of reconstruction by two methods at M = 65
(with h_1 in rectangle method)

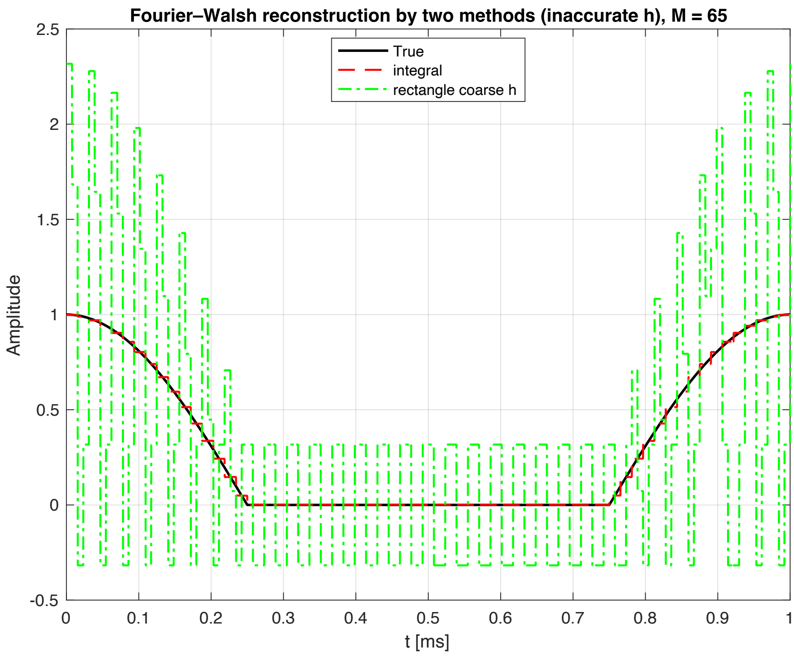

Fig.12. Comparison of reconstruction by two methods at M = 65
(with h_2 in rectangle method)

□ Calculation of the Mean-Square Errors

Because the Walsh functions $\phi_n(t)$ are orthogonal over one period, the total energy of the signal is:

$$E_0 = \int_0^T |x(t)|^2 \, \mathrm{d}t = \frac{A^2 T}{4}$$

And the mean-square error of M term approximation is:

$$\varepsilon_M = \int_0^T |x(t) - x_M(t)|^2 \, \mathrm{d}t = E_0 - T \sum_{n=0}^{M-1} |c_n|^2.$$

Similalrly as before, in errors approximation, we consider cases with both h_1 (accurate) and h_2 (inaccurate) for rectangle methods. The results are plotted on Fig.13-14.

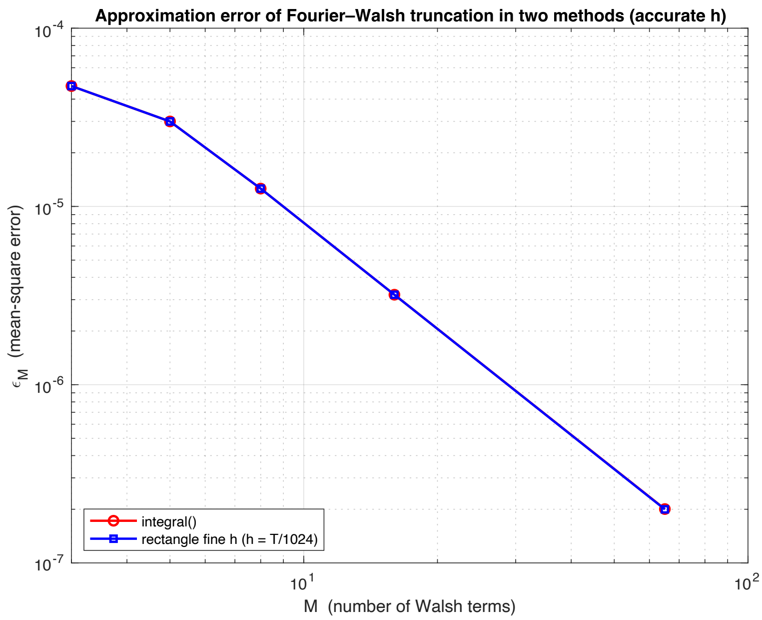

Fig.13. Approximation error of Fourier-Walsh truncation at $M = 2, 4, 8, 16, 65$ (with h_1 in rectangle method)

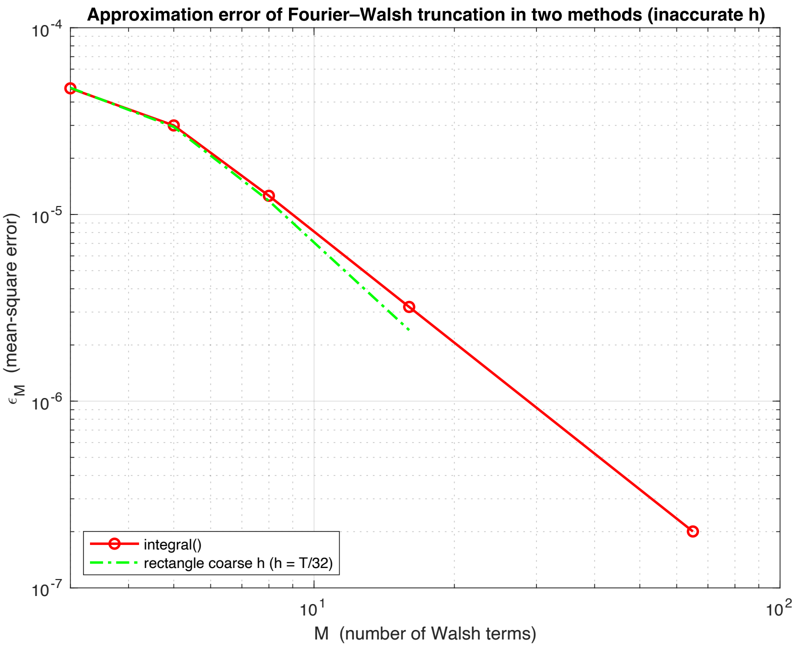

Fig.14. Approximation error of Fourier-Walsh truncation at $M = 2, 4, 8, 16, 65$

(with h_2 in rectangle method)

Discussion

1. Influence of integration step size h on accuracy of coefficients at different Walsh function order n

The numerical results show that the step size h in the rectangle method has a strong influence on the accuracy of the computed Fourier-Walsh coefficients. This behavior is also under the influence of the Walsh functions order n.

As h decreases with larger N, the absolute error decreases approximately linearly on a log scale for all n.

At low-order coefficients (e.g., $n = 3$) have larger errors compared to higher-order coefficients (e.g., $n = 11$, $n = 15$) uner the same h. However, the decrease of error from lower to higher order is not strict. For instance, we observe that the error at $n = 11$ is smaller than at $n = 15$ for the same h. This irregularity is due to the fact that the true size of each coefficient depends on the signal content, not just on n. Some higher-order coefficients may naturally carry more signal energy, making them more sensitive to integration error.

Thus, while reducing h improves the accuracy for all coefficients, the effect across different Walsh orders is not perfectly uniform, and depends also on the specific structure of the signal.

2. Observed effect of reconstructed signal $x_M(t)$

The quality of the reconstructed signal depends strongly on both the number of terms M in the truncated Fourier–Walsh sum and the accuracy of the coefficient computation. When using accurate integration (either with built-in function or a fine step h_1 in the rectangle method), the reconstruction improves with increasing M.

At lower orders (i.e. $M = 3, 5, 8, 16$), the reconstructed signals have a staircase-like shape with large steps that roughly follow the trend of the true signal, while at a very high M (i.e. $M = 65$), the approximation becomes a finer staircase-shaped with many small steps, which captures better both the amplitude and flat segments of the true signal.

In contrast, when using a coarse step h_2 in the rectangle method, the reconstruction at high M becomes unstable. The signal shows unnatural oscillations, particularly at

$M = 65$. This behavior is due to the significant errors in the higher-order coefficients, as large M includes more of them, their incorrect values introduce errors and distorting the shape and amplitude of the approximation. However, the reconstructed signal remains relatively stable even with coarse integration for low M, since low-order coefficients are less affected by the step size.

3. The influences of integration methods on the approximation errors

The mean-square error plot confirm that accurate integration methods (build-in function and rectangle method with fine step) produce errors that decreases consistently with increasing M, demonstrating the expected convergence of the Fourier-Walsh approximation. In contrast, the inaccurate rectangle method can still produce approximation errors that are close to those of build-in function method at low M, for the largest case $M = 65$, we were unable to obtain a meaningful approximation error, since the high-order coefficients become too unreliable to support a valid reconstruction.

4. Comparison of two methods

When comparing the reconstruction results and approximation errors obtained using the rectangle method and the built-in function, we notice that when a sufficiently small step size h is used, the rectangle method produces coefficients and thus reconstructions nearly identical to those from build-in function. However, the accuracy of rectangle method is dependent on the chosen integration step. Using a coarse step size in the rectangle method results in reconstructions slightly deriving from the those generated by built-in function method at low M and a large distortion at high M.

Conclusion

In this report, we investigated the reconstruction of signals using truncated Fourier–Walsh series and examined how the accuracy of the approximation depends on the number of terms and the method used to compute the coefficients. Two integration methods were compared: the rectangle method and MATLAB's built-in integral function. The results confirmed that when a fine step size is used in the rectangle method, the computed coefficients closely match those from the built-in method, and the reconstruction can be the same accurate and stable. Additionally, the approximation errors were analyzed using mean-square error plots. These confirmed that accurate integration leads to consistently decreasing errors with increasing M.

However, using a coarse step size in the rectangle method leads to significant errors in the high-order coefficients, especially for large terms, resulting in distorted reconstructions and irregular oscillations. Mean-square error analysis confirmed that coarse integration prevents proper convergence.

These findings highlight the importance of choosing an appropriate integration strategy when computing Fourier–Walsh coefficients in applications where high accuracy and stability are required.

References

[1] R. Z. Morawski, Lecture Notes for the Course *Numerical Methods*, Warsaw University of Technology, Faculty of Electronics and Information Technology, spring semester 2024/25.

[2] R. Z. Morawski, A. Miękina, *Solved Problems in Numerical Methods for Students of Electronics and Information Technology*, Oficyna Wydawnicza Politechniki Warszawskiej, 2021.

[3] Numerical methods (ENUME), Assignment B: Approximation of function, semester 2025L.

Listing of the developed program

```
clear; clc; close all;
%Global parameters
A    = 1;              % cosine amplitude
f0   = 1e3;            % frequency in Hz
w0   = 2*pi*f0;        % angular frequency
T    = 1/f0;           % period

Mlist = [3 5 8 16 65]; % truncation orders to demonstrate
Mmax  = max(Mlist); % maximum order for coefficient computation

Nvec  = 2.^(5:10);  % number of sub-intervals for rectangle rule
idxRect = numel(Nvec);  % pick finest h = T/Nvec(idxRect)

%Walsh basis preprocessing (Hadamard)
Hsize = 2^nextpow2(Mmax);
H      = hadamard(Hsize);
trans = sum(abs(diff(H,1,2)),2)/2;
[~, seqOrder] = sort(trans);
H = H(seqOrder,:);

%Part 1 : Compute coefficients c_n
x_fun = @(t) signal_xt(t,A,w0,T);
walsh = @(n,t) walsh_fun(n,t,T,H);

CN_quad = zeros(Mmax,1);
CN_rect_1 = zeros(Mmax,1);
CN_rect_2 = zeros(Mmax,1);

for n = 0:Mmax-1
    CN_quad(n+1) = (1/T) * integral(@(tt)x_fun(tt).*walsh(n,tt), ...
                               0,T,'ArrayValued',true);
end

h_1 = T/Nvec(idxRect); %the most accurate h in rectangular method
ti_1 = 0:h_1:T-h_1;
x_rect_1 = x_fun(ti_1);
```

```
for n = 0:Mmax-1
    phi = walsh(n,ti_1);
    CN_rect_1(n+1) = h_1 * sum(x_rect_1 .* phi) / T;
end

h_2 = T/Nvec(1); %the least accurate h in rectangular method
ti_2 = 0:h_2:T-h_2;
x_rect_2 = x_fun(ti_2);

for n = 0:Mmax-1
    phi = walsh(n,ti_2);
    CN_rect_2(n+1) = h_2 * sum(x_rect_2 .* phi) / T;
end

%effect of step h on coeff error
h_vec   = T./Nvec;                    % rectangle widths for all
Nvec
K       = numel(Nvec);

%compute rectangle-rule coefficients for *each* h (keep local
variable)
CN_rect_all = zeros(Mmax,K);
for k = 1:K
    h_k  = h_vec(k);
    ti   = 0:h_k:T-h_k;
    x_rk = x_fun(ti);
    for n = 0:Mmax-1
        phi                = walsh(n,ti);
        CN_rect_all(n+1,k) = h_k*sum(x_rk.*phi)/T;
    end
end

errMat = abs(CN_rect_all - CN_quad);   % |c_n^{rect}(h) -
c_n^{quad}|
errMat(errMat==0) = eps;              % avoid log(0)

%sort h_vec and errMat before plotting
[h_vec_sorted, idx_sort] = sort(h_vec, 'descend');  % h large to
small
errMat_sorted = errMat(:, idx_sort);
```

```
figure('Name','Rectangle-rule coefficient error vs h', ...
       'NumberTitle','off'); hold on

showIdx = [3 7 11 15];                % n values to display
for k = 1:numel(showIdx)
    n = showIdx(k);
    semilogy(h_vec, errMat(n+1,:),'-o','LineWidth',1.2, ...
             'DisplayName',sprintf('n = %d',n));
end

set(gca,'XGrid','on','YGrid','on');
xlabel('h  (rectangle width)');
ylabel('|c_n^{rect} - c_n^{quad}|');
title('Influence of step size h on the accuracy of coefficients');
legend('Location','eastoutside');
drawnow;


%Part 2 : Plot reconstructions for different M
Nt_plot = 4000;
t_dense = linspace(0,T,Nt_plot);
x_true  = x_fun(t_dense);

WalshMat = zeros(Mmax,Nt_plot);
for n = 0:Mmax-1
    WalshMat(n+1,:) = walsh(n,t_dense);
end

for Mm = Mlist
    %First figure: accurate reconstruction (integral + fine h)
    figure('Name',sprintf('Reconstruction (accurate) M=%d',Mm),
'NumberTitle','off');
    plot(t_dense*1e3, x_true,'k-','LineWidth',1.5); hold on;
    plot(t_dense*1e3, CN_quad(1:Mm).'*WalshMat(1:Mm,:), 'r--
','LineWidth',1.2);
    plot(t_dense*1e3, CN_rect_1(1:Mm).'*WalshMat(1:Mm,:),
'b:','LineWidth',1.2);
    xlabel('t [ms]'); ylabel('Amplitude');
    title(sprintf('Fourier-Walsh reconstruction by two methods
(accurate h), M = %d',Mm));
    legend('True','integral','rectangle fine h','Location','Best');
```

```
    grid on;

    %Second figure: inaccurate reconstruction (integral + coarse h)
    figure('Name',sprintf('Reconstruction (coarse) M=%d',Mm),
'NumberTitle','off');
    plot(t_dense*1e3, x_true,'k-','LineWidth',1.5); hold on;
    plot(t_dense*1e3, CN_quad(1:Mm).'*WalshMat(1:Mm,:), 'r--
','LineWidth',1.2);
    plot(t_dense*1e3, CN_rect_2(1:Mm).'*WalshMat(1:Mm,:),
'g-.','LineWidth',1.2);
    xlabel('t [ms]'); ylabel('Amplitude');
    title(sprintf('Fourier-Walsh reconstruction by two methods
(inaccurate h), M = %d',Mm));
    legend('True','integral','rectangle coarse
h','Location','Best');
    grid on;
end


%Part 3 : Compute & plot mean-square error ε_M
Mvec = Mlist;               % 2, 4, 8, 16   (no intermediate points)

E0   = A^2 * T / 4;         % analytical ∫|x|^2 dt
eps_quad = zeros(size(Mvec));
eps_rect_1 = zeros(size(Mvec));
eps_rect_2 = zeros(size(Mvec));

for k = 1:numel(Mvec)
    m = Mvec(k);
    eps_quad(k) = E0 - T*sum(abs(CN_quad(1:m)).^2);
    eps_rect_1(k) = E0 - T*sum(abs(CN_rect_1(1:m)).^2);
    eps_rect_2(k) = E0 - T*sum(abs(CN_rect_2(1:m)).^2);
end

%Accurate methods (integral + rectangle fine h)
figure('Name','Mean-square error vs M
(accurate)','NumberTitle','off');
loglog(Mvec, eps_quad,'ro-','LineWidth',1.5,
'DisplayName','integral()'); hold on; grid on;
loglog(Mvec, eps_rect_1,'bs-','LineWidth',1.5, ...
```

```
    'DisplayName',sprintf('rectangle fine h (h = T/%d)',
Nvec(idxRect)));
xlabel('M  (number of Walsh terms)');
ylabel('\epsilon_M  (mean-square error)');
title('Approximation error of Fourier–Walsh truncation in two
methods (accurate h)');
legend('Location','southwest');

%Inaccurate method (rectangle coarse h)
figure('Name','Mean-square error vs M (coarse
h)','NumberTitle','off');
loglog(Mvec, eps_quad,'ro-','LineWidth',1.5,
'DisplayName','integral()'); hold on; grid on;
loglog(Mvec, eps_rect_2,'g-.','LineWidth',1.5, ...
    'DisplayName',sprintf('rectangle coarse h (h = T/%d)',
Nvec(1)));
xlabel('M  (number of Walsh terms)');
ylabel('\epsilon_M  (mean-square error)');
title('Approximation error of Fourier–Walsh truncation in two
methods (inaccurate h)');
legend('Location','southwest');

%Local functions
function x = signal_xt(t,A,w0,T)
    tau = mod(t+T/2,T)-T/2;
    x   = A*cos(w0*tau).*(abs(tau)<=T/4);
end

function phi = walsh_fun(n,t,T,H)
    idx = floor(mod(t,T)/T*size(H,1)) + 1;
    phi = H(n+1,idx);
end
```