```
clear;
% Start of main_qpt.m
format long
char_precision = '%.15e';

% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
% 0. Parameters for Numerical Experiments
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
list_k    = [1, 3, 5, 7, 9, 11, 13];
list_Nrep = [100, 300, 1000, 3000, 10000, 30000, 100000];
Nave      = 400;

seed_x = 999;
gene_x = 'twister';

%
num_k    = numel(list_k);
num_Nrep = numel(list_Nrep);


% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
% 1. System Information & Estimation Setting
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

dim = 2;% 1-qubit system.
%dim = 4;% 2-qubit system.

size_Choi  = dim * dim;% size of Choi matrix, d^2 x d^2.
label = 1;
matI2 = eye(dim);

eps_sedumi = 1e-8;% = sedumi.eps, desired accuracy of optimization, used in↙
sdpsettings().

% Target Gate
theta = 0.50 * pi;
vecH_target = [0.0; 0.50 * theta ; 0.0; 0.0];

HSgb_L_target = HSgb_L_from_vecH_1qubit(vecH_target);
HSgb_G_target = expm(HSgb_L_target);

% Prepared Gate
% Hamiltonian part
vecE = [0.0; 0.010; 0.0; 0.0];
vecH_prepared = vecH_target + vecE;

% Dissipator part
T1 = 100;% us
T2 = 100;% us
alpha = 0.20;
t = 20 * power(10, -3);% us

HScb_L_prepared  = HScb_L_model_rotation_BE99_1qubit(vecH_prepared, T1/t, T2/t,↙
alpha);
HSgb_L_prepared  = HSgb_from_HScb_1qubit(HScb_L_prepared);
HSgb_G_prepared  = expm(HSgb_L_prepared);
Choi_G_prepared  = Choi_from_HSpb_1qubit(HSgb_G_prepared);

diff = norm(HSgb_G_prepared - HSgb_G_target);
diff_HSgb_G_target_prepared = diff .* diff;
display(diff_HSgb_G_target_prepared);
pause()
```

```
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
% 2. Prepared States and Prepared POVMs
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
p = 0.02;
filename_state_prepared = './ImportFiles/tester_1qubit_state_withError.csv';
num_state = FilePreparation_1qubit_state_withError(filename_state_prepared, p,↙
char_precision);

filename_povm_prepared = './ImportFiles/tester_1qubit_povm.csv';
[num_povm, num_outcome] = FilePreparation_1qubit_povm(filename_povm_prepared,↙
char_precision);

list_state_prepared    = FileImport_state(filename_state_prepared, dim, num_state);
list_povm_prepared     = FileImport_povm(filename_povm_prepared, dim, num_povm,↙
num_outcome);


% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
% 3. Tester States, Tester POVMs, IDs, Weights.
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
filename_state_tester = './ImportFiles/tester_1qubit_state.csv';
num_state = FilePreparation_1qubit_state(filename_state_tester, char_precision);

filename_povm_tester = './ImportFiles/tester_1qubit_povm.csv';
[num_povm, num_outcome] = FilePreparation_1qubit_povm(filename_povm_tester,↙
char_precision);

filename_schedule = './ImportFiles/schedule.csv';
num_schedule = FilePreparation_1qubit_schedule(filename_schedule, num_state,↙
num_povm);

filename_weight = './ImportFiles/weight_2valued_uniform.csv';
FilePreparation_1qubit_weight_2valued_uniform(filename_weight, filename_schedule,↙
char_precision);

list_state_tester = FileImport_state(filename_state_tester, dim, num_state);
list_povm_tester  = FileImport_povm(filename_povm_tester, dim, num_povm, num_outcome);
list_weight   = FileImport_weight(filename_weight, num_outcome);
list_schedule = csvread(filename_schedule);


% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
% 4. Tester States, Tester POVMs, IDs, Weights.
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

% k, Amplification information
squared_error_HSgb_G = zeros(num_k, num_Nrep, Nave);
for i_k = 1:num_k
    k = list_k(i_k);
    eps_overlap = 0.10;

    HSgb_kL_prepared = k.*HSgb_L_prepared;% used for performance evaluation later
    HSgb_Gk_prepared = expm(HSgb_kL_prepared);
    Choi_Gk_prepared = Choi_from_HSpb_1qubit(HSgb_Gk_prepared);


    % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
    % 4.1. Calculation of List of Prepared Probability Distributions
    % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
    list_probDist = ListProbDist_QPT_v2( Choi_Gk_prepared, list_state_prepared,↙
list_povm_prepared, list_schedule );
```

```
    % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
    % 4.2 Generation of List of Empirical Distributions
    % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
    for i_ave = 1:Nave
        seed_x = seed_x + i_ave;
        set_list_empiDist = set_list_empiDist_from_list_probDist_list_Nrep↙
(list_probDist, list_Nrep, seed_x, gene_x);
        num_id    = size(set_list_empiDist, 2);
        num_value = size(set_list_empiDist, 3);

        for i_Nrep = 1:num_Nrep
            list_empiDist = zeros(num_id, num_value);
            for id = 1:num_id
                for i_value = 1:num_value
                    list_empiDist(id, i_value) = set_list_empiDist(i_Nrep, id,↙
i_value);
                end
            end

            % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
            % 4.3 Calculation of D, vec(E), and h.
            % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
            matD = MatD( list_state_tester, list_povm_tester, list_schedule,↙
list_weight );
            vecE = VecE( list_state_tester, list_povm_tester, list_schedule,↙
list_weight, list_empiDist );
            h    = ConstantH( list_weight, list_empiDist );

            % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
            % 4.4 Optimization
            % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

            % 4.4.1 Standard QPT

            option            = sdpsettings('solver', 'sedumi');
            option            = sdpsettings(option, 'sedumi.eps', eps_sedumi);
            varChoi           = sdpvar(size_Choi, size_Choi, 'hermitian', 'complex');
            constraints       = [PartialTrace(varChoi, dim, dim, label) == matI2,↙
varChoi >=0];
            objectiveFunction = WeightedSquaredDistance(varChoi, matD, vecE, h);

            tic
            optimize(constraints, objectiveFunction, option);
            toc

            obj_opt  = value(objectiveFunction);
            Choi_Gk_est = value(varChoi);
            %option.sedumi

            % 4.4.2 matrix k-th root
            eps = 0.10;
            HSpb_Gk_est = HSpb_from_Choi_1qubit(Choi_Gk_est);
            nPlogMat_HSpb_Gk = logMat_nonPrincipal_1qubit(vecH_target, HSpb_Gk_est, k,↙
eps_overlap);
            HSpb_L_est = nPlogMat_HSpb_Gk ./ k;
            HSpb_G_est = expm(HSpb_L_est);
            Choi_G_est = Choi_from_HSpb_1qubit(HSpb_G_est);


            % = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
            % 5. Result Analysis
```

```
            %  = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

            % 5.1 Physicality Check
            %eig(Choi_opt)

            % 5.2 Dynamics Generator Analysis

            % 5.3 Goodness of Fit

            % 5.4 Estimation Error

            %Choi
            %Choi_Gk_est - Choi_Gk_prepared;
            %error_Choi_Gk = norm(Choi_Gk_est - Choi_Gk_prepared)
            %error_Choi_G  = norm(Choi_G_est - Choi_G_prepared)

            error_HSgb_G = norm(HSpb_G_est - HSgb_G_prepared);
            squared_error_HSgb_G(i_k, i_Nrep, i_ave) = error_HSgb_G .* error_HSgb_G;

            % 5.5 Error Bar by Bootstrap

            % 5.6 Make a report


            % End of main_lsqpt_1qubit.m
        end% i_Nrep
    end% i_Nave
end% i_k
%squared_error_HSgb_G;

% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
% 6. Output files
% = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
fileID = fopen('./OutputFiles/output_lsqpt_191207_test.txt','w');
fprintf(fileID,'%1s %4s %5s %20s\n','k', 'Nrep', 'i_ave', 'squared_error_HSgb_G');
for i_k = 1:num_k
    k = list_k(i_k);
    for i_Nrep = 1:num_Nrep
        Nrep = list_Nrep(i_Nrep);
        for i_ave = 1:Nave
            fprintf(fileID,'%d %d %d %12.11f\n', k, Nrep, i_ave, squared_error_HSgb_G↙
(i_k, i_Nrep, i_ave));
        end
    end
end
fclose(fileID);

filename = './OutputFiles/191207_squared_error_GSgb_G.mat';
save(filename, 'squared_error_HSgb_G');
```