Question 1: Cholesky identification and monetary policy shocks

In the dataset, in the first sheet of `data_ps3.xlsx`, you will find the following variables: `log(real GDP)`, `log(price deflator)`, and the federal funds rate for the U.S. from 1960 to 2010 at quarterly frequency. Estimate a VAR on those variables, identify the monetary policy shock with a triangular (Cholesky) identification structure with the ordering above, and compute impulse responses and variance decompositions for the monetary policy shock. Compute confidence intervals around the point estimates for the impulse responses using a bootstrap.

Hint 1 A bootstrap is similar to a Monte Carlo, but instead of sampling from a known distribution, you draw from the estimated residuals.

The steps are:

a. Estimate the VAR (suppose with a constant) on the N variables in y_t and store the coefficients (with the constants collected in the $(N \times 1)$ vector $\hat{c}$ and the $(N \times N)$ autoregressive coefficients in the matrix $\hat{A}$).

If there are p lags, you can express the VAR(p) as a VAR(1) with the companion form, and the $(T \times N)$ residuals $\hat{\varepsilon}$ (note that T is the number of estimated residuals, those obtained once one eliminated the lags lost in the estimation).

Solution.

We begin by importing the data and setting up the problem

```
% Settings
H = 12;      % IRF / FEVD horizon
K = 5000;    % Number of bootstrap replications

% Data Loading
file_ex1 = 'ps3/data/ps3_monetary_shock.csv';
data_ex1 = readtable(file_ex1, 'PreserveVariableNames', true);

Y = [data_ex1.log_gdp, data_ex1.log_p, data_ex1.ffr];
labels = ["log GDP", "log Price Level", "FFR"];
[n_obs, n_vars] = size(Y);
```

Let $y_t \in \mathbb{R}^N$ collect $\{\log(\text{real GDP}), \log(\text{price deflator}), \text{FFR}\}$ in that order. We can now estimate a VAR(p) in reduced form with an intercept as:

$$y_t = c + A_1 y_{t-1} + \cdots + A_p y_{t-p} + u_t, \qquad \mathbb{E}[u_t u_t'] = \Sigma_u. \tag{1.1}$$

Equation-by-equation OLS is consistent for $\{c, A_1, \ldots, A_p\}$ and yields the residual covariance Σ_u.

For computing impulse responses (and as suggested in the text of the assignment), it is convenient to stack into the VAR(1) companion form,

$$\mathbf{Y}_t = C + \mathcal{A}\mathbf{Y}t - 1 + U_t, \quad \mathbf{Y}_t = \begin{bmatrix} \mathbf{y}_t \\ \mathbf{y}_{t-1} \\ \vdots \\ \mathbf{y}_{t-p+1} \end{bmatrix}, \qquad \mathcal{A} = \begin{bmatrix} A_1 & A_2 & \cdots & A_{p-1} & A_p \\ I_k & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_k & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & I_k & \mathbf{0} \end{bmatrix}, \tag{1.2}$$

which implies a vector MA (VMA) representation underpinning impulse responses.

To identify structural (orthogonal) shocks ε_t, we impose a recursive (triangular) structure via the Cholesky factor of Σ_u. Specifically, define the $N \times N$ lower-triangular impact matrix P such that

$$u_t \;=\; P\,\varepsilon_t, \qquad \mathbb{E}[\varepsilon_t \varepsilon_t'] = I_N, \tag{1.3}$$

which delivers exact identification by setting $(N^2 - N)/2$ contemporaneous coefficients to zero. With variables ordered *(GDP, price level, FFR)*, the monetary policy shock is last and may contemporaneously move all variables, while output is contemporaneously insulated and the price level reacts contemporaneously to output but not to policy. This is the standard Sims-style recursive identification (a minimal set of assumptions) and fixes a unique mapping from reduced-form to structural shocks.

The implied structural IRFs at horizon h are

$$\Psi_h \;=\; J\,\mathcal{A}^h J'\,P, \qquad J = \begin{bmatrix} I_N & 0 \end{bmatrix}, \tag{1.4}$$

and are read columnwise by shock and rowwise by variable.

Moving on to the implementation, we choose the VAR lag length by estimating VAR(p) models for $p = 1, \ldots, 8$ and computing three systemwide information criteria, namely AIC, BIC and HQC, for each p. For a given p, let U_p be the $T \times N$ matrix of OLS residuals and $S_p = (U_p' U_p)/T$ the maximum-likelihood estimate of the residual covariance. The criterion takes the form

$$\mathrm{IC}(p) = \operatorname{logdet} S_p \;+\; \frac{c_T\, k(p)}{T}, \quad \text{with} \quad k(p) = N^2 p + N, \tag{1.5}$$

where $c_T = 2$ (AIC), $c_T = \log T$ (BIC), and $c_T = 2 \log\log T$ (HQC).

```
maxp = 8;
IC   = nan(maxp,3);   % [AIC, BIC, HQC]
models = cell(maxp,1);

for pp = 1:maxp
    m = estimateVAR(Y, pp);
    T = m.n_eff;                        % effective sample = n_obs - pp
    k = m.n_vars^2 * pp + m.n_vars;  % total parameters (system-wide)

    % ML covariance for information criteria
    S = (m.U' * m.U) / T;
    S = (S + S')/2;                     % enforce symmetry

    % robust log |S|
    [R,pflag] = chol(S);
    if pflag ~= 0
        % gentle jitter if needed
        [V,D] = eig(S);
        D = max(D, 1e-12*eye(size(D)));
        S  = V*D*V';
        R  = chol(S);
    end
    ll = 2*sum(log(diag(R)));           % = logdet(S)

    % ICs per standard VAR formulas
    IC(pp,1) = ll + (2*k)/T;            % AIC
    IC(pp,2) = ll + (log(T)*k)/T;       % BIC
    IC(pp,3) = ll + (2*log(log(T))*k)/T; % HQC

    models{pp} = m;
end

[~,pAIC] = min(IC(:,1));
```

```
[~,pBIC] = min(IC(:,2));
[~,pHQC] = min(IC(:,3));

fprintf('Selected lag orders:\n AIC: %d\n BIC: %d\n HQC: %d\n', pAIC, pBIC,
    pHQC);

p = pHQC;
```

From a theorical perspective BIC is consistent but can underfit in short samples; AIC often overfits; HQC is a common compromise. We therefore adopt the HQC-selected p (here $p = 4$).

For what it concerns the estimation of the VAR, given the repetitiveness of the task, we opt for a function-based approach by defining the four following helper functions, which we will use to estimate and store all of the required variables, namely (i) the VAR parameters, (ii) the impact matrix, (iii) the Impulse Response Functions (IRF), and (iv) the Forecast Error Variance Decompositions.

i. `estimateVAR(Y, p)`

Builds and estimates a VAR(p) with an intercept by OLS, then assembles the stacked coefficient matrix $A = [A_1 \ \cdots \ A_p]$ and the companion matrix $\mathcal{A}$. We store: coefficients (`B`), residuals (`U`), residual covariance (`SigmaU`), intercept vector (`c`), stacked A, and $\mathcal{A}$. Note that

- We trim the first p observations consistently in X and Y.
- `SigmaU` uses $df = T - k$ (unbiased); this differs from the ML scaling (T) but does not affect identification or IRFs.
- The orientation `A_k = B(rows_k,:).'` is crucial so that $y_t = c + \sum_{k=1}^{p} A_k y_{t-k} + u_t$ holds with column-vector dynamics.
- The companion matrix later lets us compute $\Phi_h = J\mathcal{A}^h J'$ efficiently.

```
function model = estimateVAR(Y, p)
    % Estimates a VAR(p) model with an intercept.
    % Y: (n_obs x n_vars) data matrix
    % p: lag order
    [n_obs, n_vars] = size(Y);

    X_lags = mlag(Y, p);
    X = [ones(n_obs, 1), X_lags];

    Y_trim = Y(p+1:end, :);
    X_trim = X(p+1:end, :);

    model.B = X_trim \ Y_trim; % OLS coefficients
    model.U = Y_trim - X_trim * model.B; % Residuals

    model.n_eff = size(Y_trim, 1);
    model.k_reg = size(model.B, 1);
    df = model.n_eff - model.k_reg;

    model.SigmaU = (model.U' * model.U) / df; % Residual covariance
    matrix

    % Store coefficients in a structured way
    model.c = model.B(1, :).'; % (n_vars x 1)

    % Build A = [A1 A2 ... Ap] with correct orientation (column-vector
     dynamics)
    A = zeros(n_vars, n_vars*p);
    for k = 1:p
```

```
        rows_k = (2 + (k-1)*n_vars) : (1 + k*n_vars); % rows in B for lag k
        A_k = model.B(rows_k, :).';                    % transpose is crucial
        A(:, (n_vars*(k-1)+1):(n_vars*k)) = A_k;
    end
    model.A = A;

    % Companion matrix
    model.A_comp = [A;
                    eye(n_vars*(p-1)), zeros(n_vars*(p-1), n_vars)];

    % Store other useful info
    model.p = p;
    model.n_vars = n_vars;
end
```

ii. `identifyCholesky(VAR_orig.SigmaU)`

Returns the lower-triangular impact matrix P such that $u_t = P\varepsilon_t$ and $\mathbb{E}[\varepsilon_t\varepsilon_t'] = I$. Under the chosen ordering (GDP $\rightarrow$ price $\rightarrow$ FFR), the policy shock can move all variables contemporaneously, while output is insulated on impact. Note that:

- We request the *lower* Cholesky factor to match the standard recursive (triangular) scheme.
- A gentle SPD "nudge" handles near-singular covariances (common with short samples/high p).
- P normalizes shocks to unit variance; sign conventions are set by the positive diagonal of P (flip a column if a different sign convention is desired for the policy shock).

```
function P = identifyCholesky(SigmaU)
    % Computes the Cholesky factor of the residual covariance matrix.
    try
        P = chol(SigmaU, 'lower');
    catch
        % Fallback for matrices that are not perfectly positive definite
        warning('SigmaU is not positive definite. Nudging to nearest SPD matrix.');
        Sig = (SigmaU + SigmaU') / 2;
        [V, D] = eig(Sig);
        D = max(D, 1e-12 * eye(size(SigmaU,1)));
        P = chol(V * D * V', 'lower');
    end
end
```

iii. `computeIRF(VAR_orig, P_orig, H)`

Computes $\Psi_h = \Phi_h P$ for $h = 0, \ldots, H$, where $\Phi_h = J\mathcal{A}^h J'$ are the reduced-form MA coefficients from the companion powers. Output is an array ($n\_vars \times n\_vars \times (H+1)$): rows = responses, columns = shocks, pages = horizons. Note that:

- $h = 0$ gives impact responses.
- Because P orthonormalizes shocks, columns of Ψ_h correspond to 1-s.d. structural shocks.

```
function IRF = computeIRF(VAR_model, P, H)
    % Computes structural impulse responses.
    % VAR_model: A struct from estimateVAR
    % P: (n_vars x n_vars) structural impact matrix
    % H: Horizon for IRFs

```

```
    [n_comp, ~] = size(VAR_model.A_comp);
    n = VAR_model.n_vars;

    IRF = zeros(n, n, H+1);
    J = [eye(n), zeros(n, n_comp - n)]; % Selection matrix

    A_pow = eye(n_comp);
    for h = 0:H
        Phi_h = J * A_pow * J'; % Reduced-form MA coefficient
        IRF(:,:,h+1) = Phi_h * P; % Structural IRF
        A_pow = A_pow * VAR_model.A_comp;
    end
end
```

iv. `computeFEVD(IRF_orig)`

Given structural IRFs Ψ_h, the n-step forecast error of variable i is $\sum_{s=0}^{n} \sum_j \Psi_{s,ij}\, \varepsilon_{j,t+s}$, so with orthonormal shocks the FEVD share for shock j equals the cumulated squared contribution of column j divided by the cumulated total. The code implements

$$\mathrm{FEVD}_{ij}(h) = \frac{\sum_{s=0}^{h} \Psi_{s,ij}^2}{\sum_{j'} \sum_{s=0}^{h} \Psi_{s,ij'}^2}, \quad h = 0, \ldots, H. \tag{1.6}$$

Note that:

- Rows sum to one (up to round-off).
- Order sensitivity is greatest at short horizons under Cholesky; this is expected and typically fades with h.

```
function FEVD = computeFEVD(IRF)
    % Computes Forecast Error Variance Decomposition from structural
    IRFs.
    % IRF: (n_vars x n_vars x H+1) array of impulse responses

    [n, ~, H1] = size(IRF);
    H = H1 - 1;

    FEVD = zeros(n, n, H1);

    % Cumulative sum of squared IRFs over horizons
    irf_sq_cumsum = cumsum(IRF.^2, 3);

    for h = 1:H1
        total_variance = sum(irf_sq_cumsum(:,:,h), 2); % Sum over
    shocks for each variable
        if any(total_variance > 0)
            FEVD(:,:,h) = irf_sq_cumsum(:,:,h) ./ total_variance;
        end
    end
end
```

At this point, it is sufficient to call the four functions using the parameters we set up in the settings, together with the ones we obtained throughout this point:

```
VAR_orig = estimateVAR(Y, p);

P_orig = identifyCholesky(VAR_orig.SigmaU);

IRF_orig = computeIRF(VAR_orig, P_orig, H);

FEVD_orig = computeFEVD(IRF_orig);
```

□

b. Sample with replacement from the estimated residuals so to form a new series of residuals $\tilde{\varepsilon}$ of dim $(T \times N)$.

Hint 2 Sample *with* replacement.

Hint 3 Sample one entire row (of dim $(1 \times N)$) of the matrix $\tilde{\varepsilon}$.

One way of doing it is generate T random integers from 1 to T (for example, call the random draw of integers `PER`) and then set $\tilde{\varepsilon} = \hat{\varepsilon}[\texttt{PER}, :]$; don't use the command `permute`.

Solution.

To place inference bands on IRFs without strong parametric assumptions, we use a residual-based bootstrap. As recommended, we draw with replacement from the estimated reduced-form residuals *rowwise* to preserve their cross-equation correlation structure, i.e.,

$$\tilde{U} \;=\; \{\, u_{t^\star} : t^\star \in \{1, \dots, T\} \text{ drawn i.i.d. with replacement} \,\}. \tag{1.7}$$

Sampling entire residual vectors $u_t' \in \mathbb{R}^N$ keeps the contemporaneous covariance of innovations intact; this is essential because the Cholesky identification relies on the (co)variance structure of the residuals. Enders recommends resampling the paired residuals $\{e_{1t}, e_{2t}\}$ together for exactly this reason.

Warning 1 (Why we're not using permute) Using `permute` on a single dimension would break the across-equation dependence and misrepresent the joint distribution.

We compute percentile bands for each IRF by taking the empirical 2.5% and 97.5% quantiles across bootstrap replications to form 95% confidence intervals.

```
fprintf('Bootstrapping IRFs for Exercise 1: K=%d, VAR(%d), H=%d\n', K, p, H
    );

IRF_draws = zeros(n_vars, n_vars, H+1, K);
FEVD_draws = zeros(n_vars, n_vars, H+1, K);

Y_initial = Y(1:p, :); % Initial conditions for simulation
U_centered = VAR_orig.U - mean(VAR_orig.U, 1);
```

□

c. Use the newly generated residuals and the estimated coefficients to construct new series:

$$\tilde{y}_t \;=\; \hat{c} \;+\; \hat{A}\,\tilde{y}_{t-1} \;+\; \tilde{\varepsilon}_t.$$

The starting values are the first values of y_t (in the case of a VAR(1), just y_1).

Solution.

Using the estimated coefficients and the bootstrapped residuals, we generate artificial data from the fitted VAR:

$$\tilde{y}_t \;=\; \hat{c} \;+\; \hat{A}_1\tilde{y}_{t-1} \;+\; \cdots \;+\; \hat{A}_p\tilde{y}_{t-p} \;+\; \tilde{u}_t, \tag{1.8}$$

starting from the observed initial conditions $\{y_1, \dots, y_p\}$. This produces a bootstrap sample $\{\tilde{y}_t\}_{t=1}^{T}$ consistent with the estimated dynamics and the empirical innovation distribution. The VMA logic then yields IRFs from the companion powers as in part a.

□

d. Estimate a VAR on the new series $\tilde{y}_t$; identify shocks and compute impulse responses and variance decompositions. Store the impulse responses and variance decompositions.

e. Repeat steps b. to d. K times (e.g., $K = 1000$).

Solution.

We implement steps c. to e. in a single parallel computing loop, so that the implementation looks like the following

```
tic
parfor k_boot = 1:K
    % (b) Resample residuals
    U_boot = U_centered(randi(VAR_orig.n_eff, VAR_orig.n_eff, 1), :);

    % (c) Simulate new data series
    Y_sim = simulateVAR(VAR_orig.B, Y_initial, U_boot);

    % (d) Re-estimate VAR on simulated data, identify, and compute IRFs/
    FEVDs
    VAR_boot = estimateVAR(Y_sim, p);
    P_boot   = identifyCholesky(VAR_boot.SigmaU);
    IRF_draws(:,:,:,k_boot)  = computeIRF(VAR_boot, P_boot, H);
    FEVD_draws(:,:,:,k_boot) = computeFEVD(IRF_draws(:,:,:,k_boot));
end
toc
```

FEVD definition and interpretation.

At horizon n, the jth variable's n-step forecast error can be written as a sum of current and lagged structural shocks with weights given by the VMA coefficients; the FEVD is the share of its forecast error variance attributable to each shock. In a recursive Cholesky scheme, the one-step FEVD of variable 1 is entirely due to shock 1; variable 2's is split between shocks 1–2; variable 3's between shocks 1–3, etc. Sensitivity to ordering is greatest at short horizon, and hence reporting FEVDs across several horizons is informative.

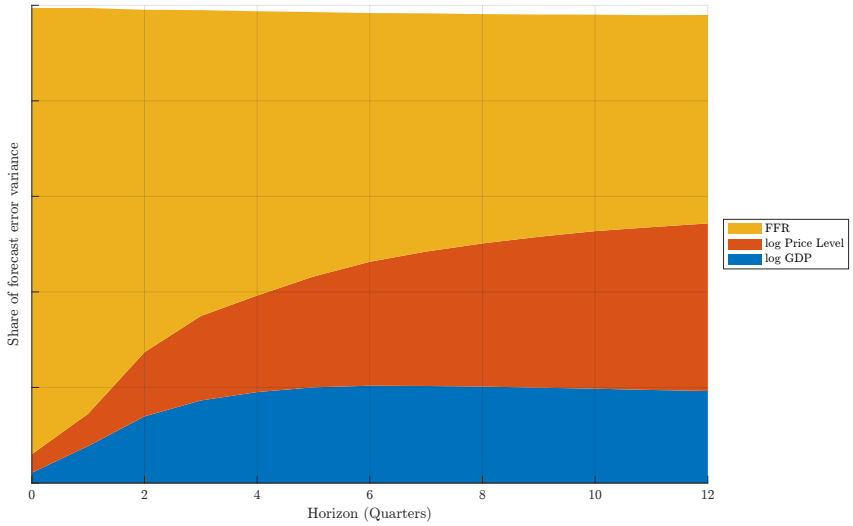
Despite not being part of the assignment, we opted for a graphical representation of the evolution of the FEVD for the three variables that we are studying, implemented through the code below:

```
shock_names = labels;                % same names for shocks and vars
t = 0:H;

% Normalize each bootstrap draw so shares sum to 1 at each (var, horizon)
FEVD_sum   = sum(FEVD_draws, 2);                      % [n x 1 x H+1 x K]
FEVD_norm  = bsxfun(@rdivide, FEVD_draws, FEVD_sum); % [n x n x H+1 x K]

% Percentiles across draws (properly normalized)
FEVD_pct   = prctile(FEVD_norm, [2.5, 50, 97.5], 4);
FEVD_med   = squeeze(FEVD_pct(:,:,:,2));  % median shares [n x n x H+1]

% Nice stacked area per variable
for i_var = 1:n_vars
    fh = figure('Position',[80 80 900 520]); hold on;
    A = squeeze(FEVD_med(i_var,:,:))';  % [H+1 x n] for area()
    area(t, A, 'LineStyle','none'); grid on;
    ylim([0 1]); yticks(0:0.2:1); yticklabels(compose('%d%%',0:20:100));
    xlim([0 H]);
    xlabel('Horizon (Quarters)');
    ylabel('Share of forecast error variance');
    title(sprintf('FEVD of %s (median across %d bootstraps)', labels(i_var)
    , size(FEVD_draws,4)));
    legend(shock_names, 'Location','eastoutside');
    set(gca,'Layer','top'); % grid above areas
```

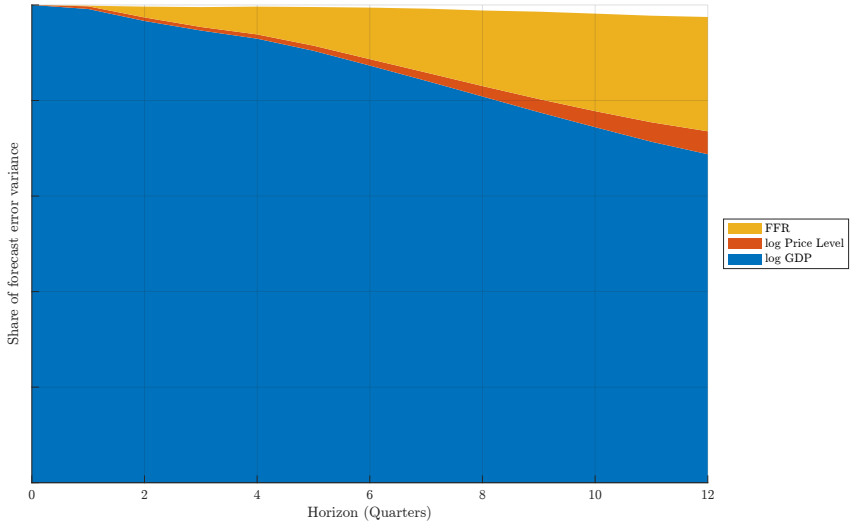
```
24
25      % Export
26      fname = sprintf('ex1_fevd_stack_%s.pdf', strrep(lower(labels(i_var)),'
        ','_'));
27      exportFig(fh, fname); close(fh);
28  end
```

On the monetary ordering used here.

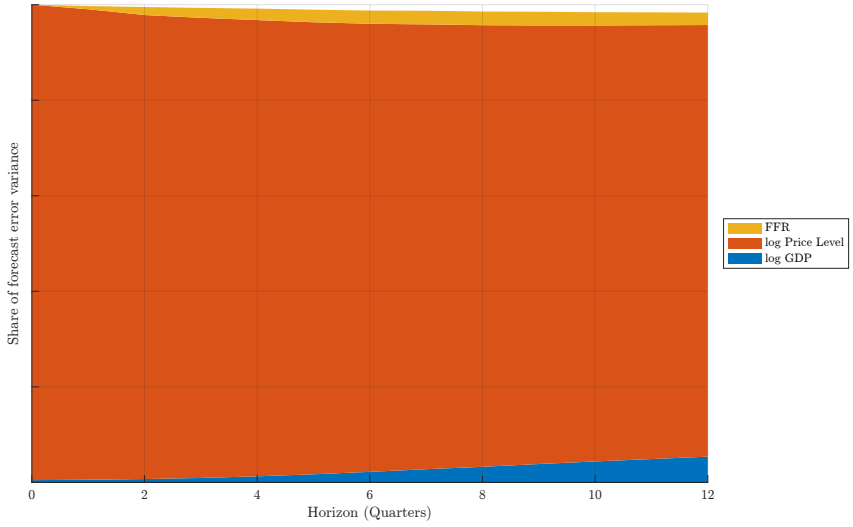
With $\{\log \text{GDP}, \log P, \text{FFR}\}$, the recursive structure implies the policy (FFR) shock is "causally last" contemporaneously—a standard assumption in monetary VARs that lets policy react within-quarter to real activity and prices, while real quantities do not jump contemporaneously to policy surprises. Enders emphasizes that such recursive identification is a minimal and transparent way to achieve exact identification; nonetheless, researchers should acknowledge that alternative plausible orderings can alter short-horizon IRFs/FEVDs when reduced-form residuals are highly correlated.



(a) FEVD of FFR



(b) FEVD of log GPD



(c) FEVD of log Price Level

Figure 1-1. Visual representations of the evolution of the FEVD (median across 5000 bootstraps).

□

f. At the end you have a set of 1000 impulse responses. Plot the 2.5% and 97.5% percentile (command `prctile`) of that empirical distribution. This is your 95% confidence interval.

Solution.

We construct point estimates, medians across draws, and 95% percentile bands (2.5–97.5). Figures plot the solid point estimate, a dashed bootstrap median, and a shaded 95% band over $H = 12$ quarters for each structural shock. This mirrors standard innovation-accounting displays and makes short- vs long-horizon uncertainty visually clear.

```
IRF_bands = prctile(IRF_draws, [2.5, 50, 97.5], 4);

for j_shock = 1:n_vars
    fh = figure('Position',[90 90 880 760]);
    tlo = tiledlayout(n_vars, 1, 'Padding', 'compact', 'TileSpacing', '
    compact');

    for i_var = 1:n_vars
        nexttile; hold on; grid on;

        % Extract IRFs for the current plot
        irf_point = squeeze(IRF_orig(i_var, j_shock, :));
        irf_lo = squeeze(IRF_bands(i_var, j_shock, :, 1));
        irf_med = squeeze(IRF_bands(i_var, j_shock, :, 2));
        irf_hi = squeeze(IRF_bands(i_var, j_shock, :, 3));

        % Plot 95% confidence band
        fill([0:H, H:-1:0], [irf_lo', fliplr(irf_hi')], ...
            [0.85 0.9 1.0], 'EdgeColor', 'none', 'FaceAlpha', 0.8, '
    DisplayName', '95% Band');

        % Plot point estimate and bootstrap median
        plot(0:H, irf_point, '-', 'LineWidth', 1.8, 'DisplayName', '
    Estimate');
        plot(0:H, irf_med, '--', 'LineWidth', 1.2, 'DisplayName', '
    Bootstrap Median');

        yline(0, 'k:', 'HandleVisibility', 'off');
        ylabel(labels(i_var));

        if i_var == 1
            title(sprintf('Responses to a "%s" Shock (Cholesky)', labels(
    j_shock)));
            legend('Location','best','Interpreter','none');
        end
        if i_var == n_vars, xlabel('Horizon (Quarters)'); end
    end

    filename = sprintf('ex1_irf_bands_shock_%s.pdf', strrep(lower(labels(
    j_shock)), ' ', '_'));
    exportFig(fh, filename);
    close(fh);
end
```

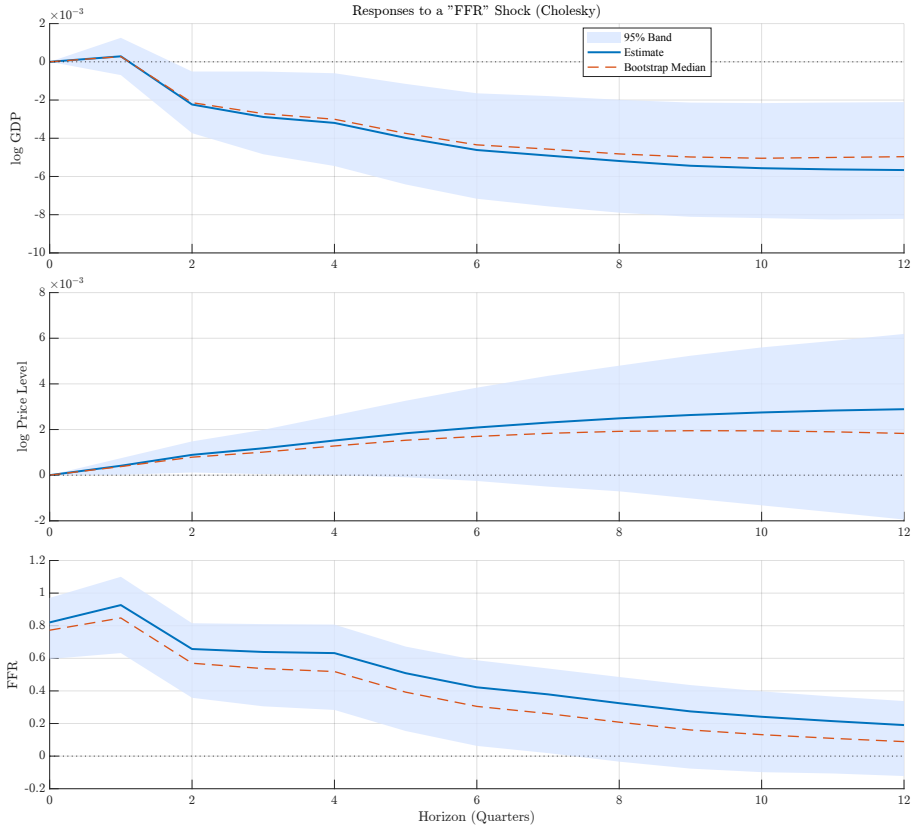


Figure 1-2. Visual representation of a response to a monetary (Federal Funds Rate) shock.

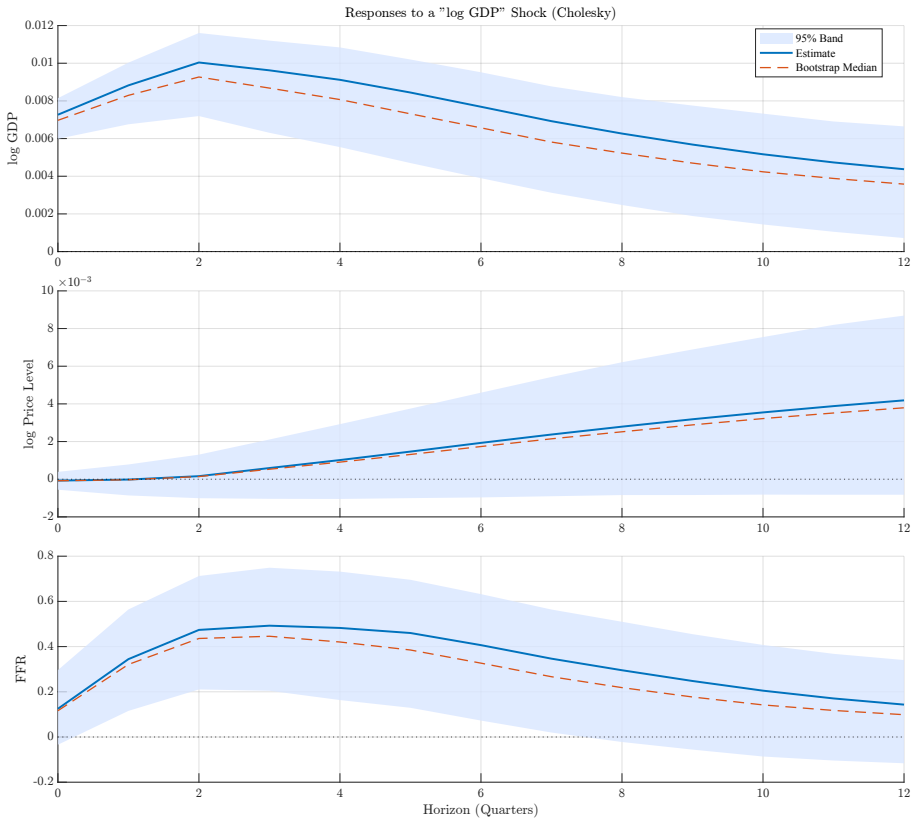


Figure 1-3. Visual representation of a response to a logGDP (output) shock.

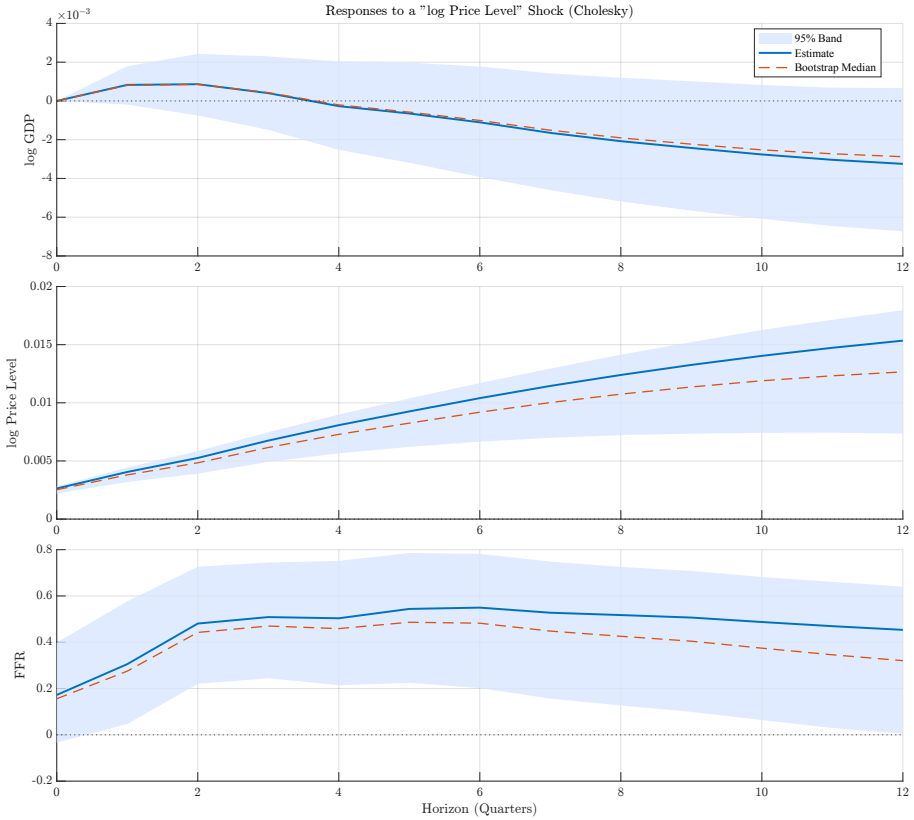


Figure 1-4. Visual representation of a response to a logPrice (price level) shock.

□

Question 2: Long-run identification

Read the paper *"Technology, Employment, and the Business Cycle: Do Technology Shocks Explain Aggregate Fluctuations?"* by Jordi Galí [1], *American Economic Review*, 89(1), 1999, 249–271.

Using the dataset in the second sheet of `data_ps3.xlsx`, replicate Figure 2 in the paper and compute bootstrapped confidence bands.

Warning 2 Results may slightly differ from those in the paper, as the data are not exactly the same.

Solution.

Following [1], I estimate a bivariate VAR in growth rates of labor productivity (Δx_t) and hours (Δn_t) and identify two orthogonal shocks: a "technology" shock and a "nontechnology" shock. The key restriction is that only the technology shock can have a permanent effect on the *level* of labor productivity. In VAR terms, this implies that the long-run impact matrix $C(1)$ is lower triangular, i.e. the cumulated effect of the nontechnology shock on x_t is zero. This is the Blanchard–Quah long-run restriction specialized to productivity and hours, and it is exactly the restriction used by Galí to disentangle technology from nontechnology shocks.

Mapping to the structural VAR. Let e_t denote VAR residuals and ε_t the structural shocks. The long-run restriction is imposed by recovering the long-horizon multipliers from the estimated VAR and choosing the contemporaneous mapping P so that $C(1)P$ is lower triangular. This follows the standard derivation linking e_t and ε_t through short-run impact and long-run multipliers in the BQ framework.

2.1 Estimating the VAR

Panel A of Figure 2 in [1] uses a bivariate system for productivity growth and first-differenced hours. I mimic that setup by constructing $Y_t = [\Delta \log(\text{Productivity})_t,\ \Delta \log(\text{Hours})_t]$ and estimating a VAR(p) with a constant. I set $p = 4$ quarterly lags, horizon $H = 12$, and compute IRFs for both identified shocks. This matches Galí's bivariate Figure 2 configuration (first-differenced hours and two-s.e. bands).

```
% Settings
p_gali = 4;
H_gali = 12;
K_gali = 1000;

% Data Prep (Gali 1999, Panel A)
file_ex2 = 'ps3/data/ps3\_technology_shock.csv';
data_ex2 = readtable(file_ex2, 'PreserveVariableNames', true);

% VAR uses growth rates: \Deltalog(Productivity), \Deltalog(Hours)
prod_level = log(data_ex2.y_l);
hours_level = log(data_ex2.hours);

Y_gali = [diff(prod_level), diff(hours_level)];
labels_gali = ["Productivity", "Hours"];
[~, n_vars_gali] = size(Y_gali);

% Estimate VAR on original data and identify
VAR_gali_orig = estimateVAR(Y_gali, p_gali);
```

The call `P_gali_orig = identifyLongRun(VAR_gali_orig)` computes the contemporaneous mapping P so that the long-run multiplier matrix for the VAR's moving-average representation is lower triangular. Economically, this enforces that the nontechnology shock has no permanent effect on productivity, while allowing permanent effects on hours and output (as in Galí's specification).

```
P_gali_orig = identifyLongRun(VAR_gali_orig);
IRF_gali_orig = computeIRF(VAR_gali_orig, P_gali_orig, H_gali);
```

2.2 Bootstrapped confidence bands

We use a residual (i.i.d.) bootstrap to quantify parameter uncertainty in the IRFs. Concretely:

i. center and resample the estimated VAR residuals with replacement;

ii. simulate pseudo-data from the estimated VAR using those resampled shocks and the observed initial conditions;

iii. re-estimate the VAR and re-identify the shocks in each draw;

iv. recompute IRFs and take empirical percentiles across draws. This is the standard residual-bootstrap approach for IRF confidence intervals (closely related to the Monte Carlo bands discussed in Enders).

Galí reports two-s.e. bands generated from 500 Monte Carlo draws, while I use $K = 1000$ bootstrap replications. Minor differences in band width are expected.

```
% Bootstrap confidence intervals
fprintf('Bootstrapping IRFs for Exercise 2: K=%d, VAR(%d), H=%d\n', K_gali,
    p_gali, H_gali);

IRF_gali_draws = zeros(n_vars_gali, n_vars_gali, H_gali+1, K_gali);
Y_gali_initial = Y_gali(1:p_gali, :);
```

```

U_centered_gali = VAR_gali_orig.U - mean(VAR_gali_orig.U,1);

tic
parfor k_boot = 1:K_gali
    % Resample residuals
    U_boot = U_centered_gali(randi(VAR_gali_orig.n_eff, VAR_gali_orig.n_eff, 1)
    , :);

    % Simulate new data series
    Y_sim = simulateVAR(VAR_gali_orig.B, Y_gali_initial, U_boot);

    % Re-estimate VAR, identify, and compute IRFs
    VAR_boot = estimateVAR(Y_sim, p_gali);
    P_boot = identifyLongRun(VAR_boot);
    IRF_gali_draws(:,:,:,k_boot) = computeIRF(VAR_boot, P_boot, H_gali);
end
toc
```

Because the VAR is in growth rates, I cumulate the IRFs over the horizon to obtain level responses. I also report the response of (log) output as the identity $y_t = x_t + n_t$, so $\Delta y_t = \Delta x_t + \Delta n_t$; hence the IRF of output growth equals the sum of the productivity and hours IRFs, and its level response is the cumulated sum of that composite. This matches the three-row layout in Galí's figures (productivity, output, hours).

```
% Transform IRFs to cumulative levels
    % Function to convert growth rate IRFs to cumulative levels for plotting
cumulate_irfs = @(irf_draws) cumsum(irf_draws, 3);

    % Calculate cumulative IRFs for output (productivity + hours)
irf_output_orig = IRF_gali_orig(1,:,:) + IRF_gali_orig(2,:,:);
irf_output_draws = IRF_gali_draws(1,:,:,:) + IRF_gali_draws(2,:,:,:);

    % Combine all variables for plotting: [Prod, Hours, Output]
IRF_plot_orig = cumulate_irfs([IRF_gali_orig; irf_output_orig]);
IRF_plot_draws = cumulate_irfs([IRF_gali_draws; irf_output_draws]);

labels_plot_gali = ["Productivity", "Hours", "Output"];
plot_titles_gali = ["Technology Shock", "Non-Technology Shock"];
n_plot_vars = length(labels_plot_gali);
```

```
% Plotting (Figure 2 Style)
IRF_plot_bands = prctile(IRF_plot_draws, [2.5, 97.5], 4);
fh = figure('Position',[50 50 980 760]);
tlo = tiledlayout(n_plot_vars, n_vars_gali, 'Padding', 'compact', 'TileSpacing'
    , 'compact');

for i_var = 1:n_plot_vars
    for j_shock = 1:n_vars_gali
        nexttile; hold on; grid on;
        irf_point = squeeze(IRF_plot_orig(i_var, j_shock, :));
        irf_lo = squeeze(IRF_plot_bands(i_var, j_shock, :, 1));
        irf_hi = squeeze(IRF_plot_bands(i_var, j_shock, :, 2));
        fill([0:H_gali, H_gali:-1:0], [irf_lo', fliplr(irf_hi')], ...
            [0.85 0.9 1.0], 'EdgeColor','none', 'FaceAlpha', 0.7);
        plot(0:H_gali, irf_point, 'k-', 'LineWidth', 1.6);
        yline(0, 'k--');
        title(sprintf('Response of %s', labels_plot_gali(i_var)));
        ylabel('\% Deviation');
        if i_var == n_plot_vars, xlabel('Quarters'); end
        if i_var == 1, title(plot_titles_gali(j_shock)); end
    end
end
```

```
22
23 title(tlo, sprintf('Replication of Gali (1999) Figure 2 | VAR(%d), K=%d',
       p_gali, K_gali));
24 exportFig(fh, sprintf('ex2_gali1999_replication_VAR%d_K%d.pdf', p_gali, K_gali)
       );
25 close(fh);
```

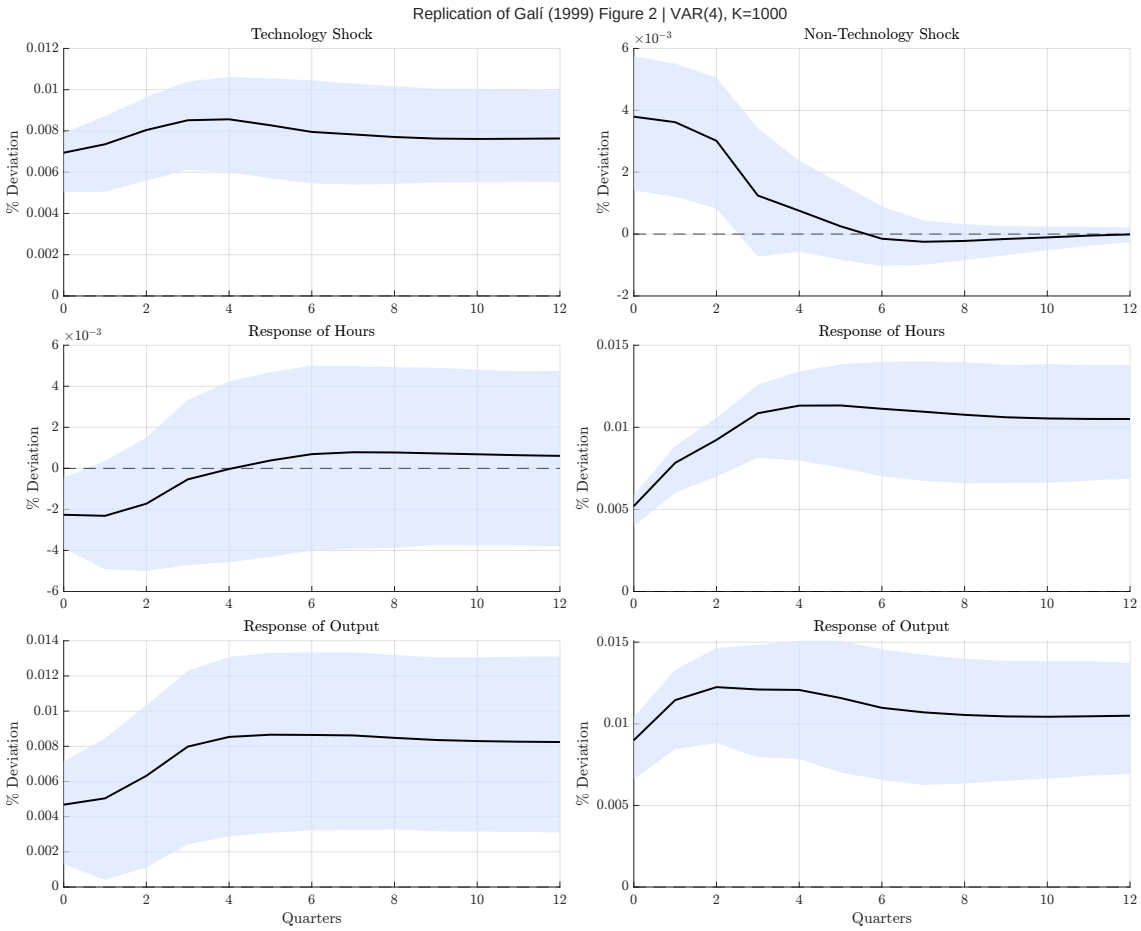


Figure 2-1. Estimated impulse responses from a bivariate model: U.S. data, first-differenced hours (point estimates and 2 standard error confidence intervals)

Consistent with [1], a positive technology shock raises labor productivity on impact and persistently, while hours fall on impact (a short-lived but notable decline), and output rises more gradually. In contrast, a positive nontechnology shock pushes up hours and output persistently, with the effect on productivity fading to zero by construction of the identifying restriction. This pattern—negative conditional comovement of hours and productivity for technology shocks and positive conditional comovement for nontechnology shocks—matches Galí's bivariate evidence. □

References

[1] Jordi Galí. "Technology, Employment, and the Business Cycle: Do Technology Shocks Explain Aggregate Fluctuations?" In: *American Economic Review* 89.1 (Mar. 1999), pp. 249–271. DOI: 10.1257/aer.89.1.249. URL: https://www.aeaweb.org/articles?id=10.1257/aer.89.1.249.