cbset sympycheckstyle/.style=colback=gray!5,colframe=black!60,leftrule=2pt,arc=1mm,boxrule=0.5pt,
leanproofstyle/.style=colback=blue!4,colframe=black!60,leftrule=2pt,arc=1mm,boxrule=0.5pt,
didacticstyle/.style=colback=green!4,colframe=black!60,leftrule=2pt,arc=1mm,boxrule=0.5pt,
mathstyle/.style=colback=gray!6,colframe=black!60,leftrule=2pt,arc=1mm,boxrule=0.5pt,
literaturestyle/.style=colback=gray!10,colframe=black!60,leftrule=2pt,arc=1mm,boxrule=0.5pt

heoremstyleplain heoremstyledefinition

A Appendix A: Detailed Derivations

A.1 HJB Analysis: Hamiltonian, KKT, and Convexity

We analyze the maximization problem embedded in the HJB equation (4). Define the optimized Hamiltonian $\mathcal{H}(k, z, x, m, p)$ with $p = V_k$:

$$\mathcal{H}(k, z, x, m, p) = \max_{i \in \mathbb{R}} \Big\{ \pi(k, i, z, x, m) + p\,(i - \delta k) \Big\}. \tag{1}$$

Substituting the profit function from **??** yields

$$\mathcal{H} = P(Y) e^{x+z} k^{\alpha} - f - p\delta k + \max_{i \in \mathbb{R}} \Big\{ (p-1)i - h(i, k) \Big\},$$

so the maximization term is the Legendre–Fenchel transform of the convex adjustment cost h evaluated at $p - 1$.

Optimal Policy Derivation (KKT/FOC). Because h is convex in i, the objective is concave. The first-order condition (FOC) is

$$p - 1 - h_i(i, k) = 0 \quad \Longrightarrow \quad i^* = h_i^{-1}(p - 1).$$

For the asymmetric quadratic cost $h(i, k) = \frac{\phi_{\pm}}{2} i^2/k$ from **??** we obtain

- $i \geq 0$: $h_i = \phi_+ i/k$. The FOC gives $p - 1 = \phi_+ i/k$ so $i = k(p-1)/\phi_+$, which is feasible only when $p \geq 1$.
- $i < 0$: $h_i = \phi_- i/k$. The FOC gives $p - 1 = \phi_- i/k$ so $i = k(p-1)/\phi_-$, which requires $p < 1$.

This recovers the piecewise-affine policy in **??**.

Verification: FOCs for the optimal policy

See Appendix E, Check 2, for a symbolic confirmation of the formulas above.

Formalization: Optimization structure in Lean 4

```
import Mathlib.Analysis.Convex.Basic

-- Parameters: k > 0, phi_plus > 0, phi_minus > 0.
variables {k phi_plus phi_minus p : Real} (hk : k > 0) (hp_p : phi_plus > 0) (hp_m :
↪  phi_minus > 0)

-- Asymmetric quadratic adjustment cost.
def adjustment_cost (i : Real) : Real :=
  if i >= 0 then (phi_plus / 2) * (i ^ 2 / k) else (phi_minus / 2) * (i ^ 2 / k)

-- Objective maximized inside the Hamiltonian.
def objective (i : Real) : Real := (p - 1) * i - adjustment_cost i

-- Candidate optimal policy i*(p).
def optimal_policy : Real :=
  if p >= 1 then k * (p - 1) / phi_plus else k * (p - 1) / phi_minus

-- TODO: Show that `optimal_policy` maximizes `objective` using convexity + FOCs.
```

A.2 Deriving the generalized master equation

We outline the stationary Generalized Master Equation (ME) (I). When the HJB coefficients $r(m)$ and $\mathcal{L}_x^{\mathbb{Q}}(m)$ depend on m, the standard derivation must be enhanced to accommodate interactions in the coefficients (Bensoussan et al., 2013).

Functional Itô calculus. Consider $U(k_t, z_t, x, m_t)$. The generalized functional Itô lemma tracks the evolution of U under changes in both the state (k_t, z_t) and the law m_t. Conditioning on x and using the pricing measure $\mathbb{Q}$, stationarity requires $r(m)U$ to balance all drift contributions.

Decomposing the m-dependence. Two channels link the law m to valuations:

(i) **Transport.** Variations of m feed through the Lions derivative $\delta_m U$ even if coefficients are fixed.

(ii) **Externalities.** Variations of m alter π, r, and $\mu_x^{\mathbb{Q}}$, feeding back into the HJB residual itself.

Applying the generalized chain rule produces

$$
\begin{aligned}
E_{\text{transport}} &= \int \Big[(i^*(\xi, m) - \delta\kappa)\, \partial_\kappa \delta_m U(\xi; \cdot) + \mathcal{L}_z \delta_m U(\xi; \cdot) \Big]\, m(\mathrm{d}\xi), \\
E_{\text{price}} &= \int \delta_m \pi(\xi; k, z, x, m)\, m(\mathrm{d}\xi), \\
E_{\text{pricing}} &= \int \Big[-(\delta_m r(\xi; m))U + (\delta_m \mu_x^{\mathbb{Q}}(\xi; m))U_x \Big]\, m(\mathrm{d}\xi).
\end{aligned}
$$

Sign intuition

If a perturbation of m raises the short rate ($\delta_m r > 0$), valuations fall, hence the negative sign. Conversely, if the perturbation raises the risk-neutral drift ($\delta_m \mu_x^{\mathbb{Q}} > 0$) the asset gains value when $U_x > 0$, giving a positive contribution.

Collecting the HJB drift, transport, and externality terms and imposing the pricing-measure martingale condition delivers the ME (I).

Assumption 1 (Primitives and environment). *1.* ***Capital law:*** $dk_t = (i_t - \delta k_t)\mathrm{d}t$, $i \in \mathbb{R}$.

2. ***Irreversibility/adjustment:*** *h convex and asymmetric,*

$$
h(i, k) = \begin{cases} \frac{\phi_+}{2} \dfrac{i^2}{k}, & i \geq 0, \\ \frac{\phi_-}{2} \dfrac{i^2}{k}, & i < 0,\ \phi_- > \phi_+. \end{cases}
$$

3. ***Dividends:*** $\pi(k, i, z, x, m) = P(Y)\, e^{x+z} k^\alpha - i - h(i, k) - f$.

4. ***Shocks (Physical measure $\mathbb{P}$):*** $dz_t = \mu_z(z_t)\mathrm{d}t + \sigma_z \mathrm{d}W_t$, $dx_t = \mu_x(x_t)\mathrm{d}t + \sigma_x \mathrm{d}B_t$ *(independent).*

5. ***Consumer Preferences and GE Closure:*** *A representative agent maximizes lifetime utility* $\mathbb{E}^{\mathbb{P}}\left[\int_0^\infty e^{-\rho t} \frac{C_t^{1-\gamma}}{1-\gamma} dt\right]$. *Goods market clearing implies C_t equals aggregate dividends D_t.*

6. ***Generators (under $\mathbb{P}$):*** *for smooth u,*

$$\mathcal{L}_z u = \mu_z(z)\, u_z + \tfrac{1}{2}\sigma_z^2 u_{zz}, \qquad \mathcal{L}_x u = \mu_x(x)\, u_x + \tfrac{1}{2}\sigma_x^2 u_{xx}.$$

Assumption 2 (Minimal regularity/boundary). *(a) $h(\cdot, k)$ convex, lower semicontinuous; $h(i,k) \geq c\, i^2/k$ for some $c > 0$ on $k > 0$.*

(b) P Lipschitz on compact sets with $P' < 0$.

(c) μ_z, μ_x locally Lipschitz; $\sigma_z, \sigma_x \geq 0$ constants.

(d) Boundary at $k = 0$: *reflecting; $i^*(0,\cdot) \geq 0$; and $U_k(0,\cdot) \leq 1$.*

(e) Growth: *$U(k,z,x,m) = O(k)$ as $k \to \infty$.*

(f) Integrability: *m integrates k^α and $1/k$.*

(g) Pricing Regularity: *Equilibrium aggregate consumption $C(x)$ is sufficiently regular (e.g., C^2) to admit well-defined dynamics via Itô's lemma (see **??**).*

Economic reading. The convex asymmetry $\phi_- > \phi_+$ produces *investment bands*. Aggregation operates through the product market (Y) and the financial market (via the SDF). The inverse-demand slope $P'(Y)$ and the endogenous market price of risk $\lambda(x,m)$ are the channels through which the cross-section affects an individual firm's valuation and decisions.

Where this sits. Zhang (2005) emphasizes how costly reversibility shapes asset prices, using an exogenous SDF. The present mean-field formulation adds an equilibrium product price mapping and a fully endogenous SDF, leading to a master PDE that makes the cross-sectional feedback explicit. For master equations, see Cardaliaguet–Delarue–Lasry–Lions (2019). The generalization for interactions in coefficients (required for the pricing externalities) follows Bensoussan, Frehse, and Yam (2013).

B Mathematical Setup: State Space, Measures, and Differentiation on $\mathcal{P}$

B.1 State space and probability metrics

We consider the state space $S \equiv \mathbb{R}_+ \times \mathbb{R}$. The population law $m \in \mathcal{P}(S)$. We restrict to the W_2-finite set $\mathcal{P}_2(S)$, metrized by the quadratic Wasserstein distance W_2.

Definition 1 (Lions derivative). *Let $F : \mathcal{P}_2(S) \to \mathbb{R}$. The* Lions derivative *$\delta_m F(m) : S \to \mathbb{R}^{d_s}$ is defined by lifting F to a function $\tilde{F}$ on $L^2(\Omega; S)$ and taking the Fréchet derivative $D\tilde{F}$.*

We use $\delta_m F(\xi; m)$ to denote the Lions derivative of a functional $F(m)$ evaluated at ξ. When we write $\delta_m U(\xi; k, z, x, m)$, we identify the derivative of $m \mapsto U(k,z,x,m)$ at point $\xi \in S$.

Lemma 1 (Chain rule for composite functionals). *Let $F(m) = G(\Phi(m))$ with $G : \mathbb{R} \to \mathbb{R}$ differentiable and $\Phi(m) = \int \varphi(\xi)\, m(\mathrm{d}\xi)$. Then $\delta_m F(m)(\xi) = G'(\Phi(m))\, \varphi(\xi)$.*

Application to the price externality. With $\varphi(\xi) = e^{x+\zeta}\kappa^\alpha$ and $G = P$, Lemma **??** yields $\delta_m\big(P(\Phi(m))\big)(\xi) = P'(Y)\, e^{x+\zeta}\kappa^\alpha$. Multiplying by the *this-firm* factor $e^{x+z}k^\alpha$ produces the integrand of the price externality term in the ME.

B.2 Generators, domains, and adjoints

The generator $\mathcal{L}_z$ acts on $C_b^2(\mathbb{R})$ functions of z. The adjoint $\mathcal{L}_z^*$ acts on densities $m(k,z)$ as $\mathcal{L}_z^* m = -\partial_z(\mu_z m) + \frac{1}{2}\sigma_z^2 \partial_{zz} m$. The transport in k is first-order; the adjoint contributes $-\partial_k\big[(i^* - \delta k)m\big]$.

C General Equilibrium and Asset Pricing

The economy is closed by the representative consumer (**??**).

Definition 2 (Aggregate Dividends and Consumption)**.** *Aggregate dividends are:*

$$D(m,x) = \int \pi(k, i^*(k,z,x,m), z, x, m)\, m(\mathrm{d}k, \mathrm{d}z).$$

In equilibrium, the goods market clears: $C(m,x) = D(m,x)$.

The stochastic discount factor (SDF), Λ_t, is determined by the consumer's marginal utility: $\Lambda_t = e^{-\rho t} C_t^{-\gamma}$.

Assumption 3 (Equilibrium Consumption Dynamics)**.** *We assume that in a stationary equilibrium, the distribution m is a function of the aggregate state x, $m(x)$. Aggregate consumption $C(x) = C(m(x), x)$ is sufficiently smooth such that its dynamics under $\mathbb{P}$ are:*

$$\frac{\mathrm{d}C_t}{C_t} = \mu_C(x_t)\mathrm{d}t + \sigma_C(x_t)\mathrm{d}B_t.$$

where by Itô's lemma:

$$\mu_C(x) = \frac{1}{C(x)}(\mathcal{L}_x C)(x) = \frac{C'(x)}{C(x)}\mu_x(x) + \frac{1}{2}\frac{C''(x)}{C(x)}\sigma_x(x)^2,$$
$$\sigma_C(x) = \frac{C'(x)}{C(x)}\sigma_x(x).$$

Proposition 1 (Endogenous Short Rate and Market Price of Risk)**.** *The equilibrium short rate $r(x)$ and the market price of risk $\lambda(x)$ for the aggregate shock B_t are given by the Lucas pricing (CCAPM) formulas:*

$$r(x) = \rho + \gamma\mu_C(x) - \frac{1}{2}\gamma(\gamma+1)\sigma_C(x)^2,$$
$$\lambda(x) = \gamma\sigma_C(x).$$

The SDF dynamics are $\frac{\mathrm{d}\Lambda_t}{\Lambda_t} = -r(x_t)\mathrm{d}t - \lambda(x_t)\mathrm{d}B_t$.

Proof. Follows from applying Itô's lemma to Λ_t and matching drift and diffusion terms. See Appendix E for verification. □

Endogeneity and Feedback. Crucially, $r(x)$ and $\lambda(x)$ depend on the equilibrium consumption dynamics, which in turn depend on the cross-sectional distribution $m(x)$ and its sensitivity to x. This introduces a general equilibrium feedback channel. We denote this dependence as $r(x,m)$ and $\lambda(x,m)$ when analyzing the general mean-field system before imposing the stationary equilibrium structure $m(x)$.

D Firm Problem and the Stationary HJB

The firm maximizes the present value of dividends discounted by the endogenous SDF Λ. This valuation is performed under the risk-neutral measure $\mathbb{Q}$, obtained via Girsanov's theorem using the market price of risk $\lambda(x, m)$.

The risk-neutral drift of x is:

$$\mu_x^{\mathbb{Q}}(x, m) = \mu_x(x) - \sigma_x(x)\lambda(x, m). \tag{2}$$

The risk-neutral generator $\mathcal{L}_x^{\mathbb{Q}}$ is:

$$\mathcal{L}_x^{\mathbb{Q}} V = \mu_x^{\mathbb{Q}} V_x + \frac{1}{2}\sigma_x^2 V_{xx}. \tag{3}$$

We assume idiosyncratic risk z is diversifiable and its generator L_z remains unchanged under $\mathbb{Q}$.

The stationary HJB equation is

$$\boxed{r(x, m)\, V = \max_{i \in \mathbb{R}} \left\{ \pi(k, i, z, x, m) + V_k\,(i - \delta k) + \mathcal{L}_z V + \mathcal{L}_x^{\mathbb{Q}} V \right\}} \tag{HJB}$$

The interior first-order condition reads

$$0 = \partial_i \pi + V_k = -(1 + h_i(i, k)) + V_k \quad \Longrightarrow \quad i^*(k, z, x, m) = h_i^{-1}\big(V_k - 1\big).$$

Proposition 2 (Explicit policy under asymmetric quadratic cost)**.** *For the adjustment cost in* **??***, the optimal policy is*

$$i^*(k, z, x, m) = \begin{cases} \dfrac{k}{\phi_+}\,\big(V_k - 1\big), & V_k \geq 1, \\ \dfrac{k}{\phi_-}\,\big(V_k - 1\big), & V_k < 1. \end{cases}$$

Economic intuition (expanded).

- *Investment bands.* The kink at $V_k = 1$ creates inaction; asymmetry ($\phi_- > \phi_+$) makes disinvestment less responsive.
- *Cyclicality and Asset Pricing.* Valuation depends on both cash flows (via $P(Y)$) and the endogenous SDF. The countercyclical market price of risk $\lambda(x)$ (Zhang 2005) implies that in recessions (low x), risk premia are high ($\mu_x^{\mathbb{Q}}$ is low), depressing firm values, particularly for firms constrained by costly reversibility.
- *Decomposition.* V_k aggregates (i) private technology and costs via the Hamiltonian, and (ii) the *general-equilibrium wedges* from inverse demand and asset pricing, made explicit in the ME via externalities.

Mathematical rigor (expanded).

- *Convexity.* The Hamiltonian $\mathcal{H}(k, z, x, m, p) \equiv \max_i\{\pi + p\,(i - \delta k)\}$ is convex in $p = V_k$.
- *Risk-Neutral Valuation.* The HJB equation represents the valuation under $\mathbb{Q}$. The dependence of r and $\mathcal{L}_x^{\mathbb{Q}}$ on m reflects the GE feedback.
- *Boundary conditions.* Reflecting at $k = 0$ implies $i^*(0, \cdot) \geq 0$ and $U_k(0, \cdot) \leq 1$.

E Kolmogorov–Forward (FP) Equation

The evolution of the population law m_t occurs under the physical measure $\mathbb{P}$. Given x and the policy i^*, m_t satisfies

$$\partial_t m = -\frac{\partial}{\partial k}\left(\left(i^\star(k,z,x,m) - \delta k\right) m\right) + \mathcal{L}_z^* m \tag{FP}$$

In stationary equilibrium conditional on x: $\partial_t m = 0$.

E.1 Boundary and integrability

Reflecting at $k = 0$ implies zero probability flux: $\left[(i^* - \delta k)m\right]\big|_{k=0} = 0$.

Degenerate transport in k. The k-direction is purely hyperbolic. Schemes must be *upwind* in k and *conservative* to maintain $\int m = 1$.

F Market Clearing

Aggregate quantity and price are

$$Y(m,x) = \int e^{x+z} k^\alpha \, m(\mathrm{d}k, \mathrm{d}z), \qquad P = P(Y(m,x)), \quad P' < 0.$$

In the isoelastic case $P(Y) = Y^{-\eta}$ with $\eta > 0$,

$$Y\,P'(Y) = -\eta\,P(Y). \tag{4}$$

Economic content. The aggregation wedge in firm incentives comes from two sources. First, the *marginal-revenue* term (price externality). Second, the *SDF* determined by aggregate consumption (pricing externality).

G Master Equation (Stationary, Conditional on x)

Define the master value $U(k,z,x,m)$ and the Lions derivative $\delta_m U(\xi; k,z,x,m)$ at $\xi = (\kappa, \zeta)$. The stationary master equation characterizes the equilibrium value U. Because the coefficients of the HJB ($r(m)$ and $\mathcal{L}_x^{\mathbb{Q}}(m)$) depend on m, we must use the generalized formulation (Bensoussan et al., 2013).

$$
\begin{aligned}
r(x,m)\,U(k,z,x,m) = \; & \underbrace{\max_i \left\{\pi(k,i,z,x,m) + U_k(i-\delta k) + \mathcal{L}_z U + \mathcal{L}_x^{\mathbb{Q}} U\right\}}_{\text{Own-firm HJB terms (under } \mathbb{Q})} \\
& + \underbrace{\int \Big[(i^*(\xi,x,m)-\delta\kappa)\,\partial_\kappa \delta_m U(\xi;\cdot) + \mathcal{L}_z \delta_m U(\xi;\cdot)\Big]\, m(\mathrm{d}\xi)}_{\text{Population transport (under } \mathbb{P}) \text{ via } \delta_m U} \\
& + \underbrace{\int \delta_m \pi(\xi;\, k,z,x,m)\, m(\mathrm{d}\xi)}_{\text{Direct price externality}} \\
& + \underbrace{\int \Big[-(\delta_m r(\xi;x,m))U + (\delta_m \mu_x^{\mathbb{Q}}(\xi;x,m))U_x\Big]\, m(\mathrm{d}\xi)}_{\text{Pricing externality}}.
\end{aligned}
\tag{ME}
$$

where $\delta_m U(\xi;\cdot)$ abbreviates $\delta_m U(\xi;k,z,x,m)$, and the transport term integrates the action of the physical generators on $\delta_m U$.

Interpreting the Master Equation. ?? decomposes the general-equilibrium effects:

(i) **HJB terms:** Optimization by the representative firm under $\mathbb{Q}$, taking prices (P, r, λ) as given.

(ii) **Population transport:** The evolution of m under $\mathbb{P}$ moves U through $\delta_m U$.

(iii) **Price externality:** Impact of m on dividends π via the product price $P(Y)$.

(iv) **Pricing externality:** Impact of m on the valuation operators (discount rate r and risk-neutral drift $\mu_x^{\mathbb{Q}}$) via the SDF.

G.1 Derivation of the General Equilibrium Externalities

Proposition 3 (General Equilibrium Externalities)**.** *The GE externalities consist of the price externality (E_{price}) and the pricing externality ($E_{pricing}$).*

Price Externality. *Let $\chi(k,z,x) = e^{x+z}k^\alpha$. The integrated price externality is:*

$$E_{price} = \chi(k,z,x)\, Y(m,x)\, P'(Y(m,x)).$$

If $P(Y) = Y^{-\eta}$, $E_{price} = -\eta\, P(Y)\, \chi(k,z,x)$.

Pricing Externality. *The pricing externality term is:*

$$E_{pricing} = \int \Big[-(\delta_m r(\xi;x,m))U + (\delta_m \mu_x^{\mathbb{Q}}(\xi;x,m))U_x\Big]\, m(\mathrm{d}\xi).$$

Since $\mu_x^{\mathbb{Q}} = \mu_x - \sigma_x \lambda$, we have $\delta_m \mu_x^{\mathbb{Q}} = -\sigma_x \delta_m \lambda$.

Proof. The price externality derivation follows from **??**. The pricing externality arises from the generalized Master Equation formulation for MFGs with interactions in the coefficients (see Appendix A.3). □

Complexity of Pricing Externalities. The terms $\delta_m r$ and $\delta_m \lambda$ depend on the Lions derivatives of the consumption dynamics $(\mu_C(m), \sigma_C(m))$. Calculating these requires understanding how the equilibrium consumption process $C(m)$ responds to infinitesimal changes in the distribution m. This involves differentiating through the fixed-point definition of the equilibrium, which is highly complex.

extbfVerification: Proposition **??** (Price Externality Derivation)

```
import sympy as sp

# Symbols
Y, eta = sp.symbols('Y eta', positive=True)
k, z, x = sp.symbols('k z x', real=True)
alpha = sp.symbols('alpha', positive=True)

# Production function and price
chi = sp.exp(x + z) * k**alpha
P = sp.Function('P')(Y)

# Integrated externality structure
integrated_externality = chi * sp.diff(P, Y) * Y

# Isoelastic demand
P_iso = Y**(-eta)
externality_iso = sp.simplify(integrated_externality.subs(P, P_iso))
target_iso = -eta * P_iso * chi

assert sp.simplify(externality_iso - target_iso) == 0
print("Proposition 8.1 price externality structure verified.")
```

G.2 Equivalence and Monotonicity

Theorem 1 (Equivalence of HJB–FP and ME formulations)**.** *Assume* **????** *hold. Under sufficient differentiability of U and $\delta_m U$, and appropriate generalized monotonicity conditions on the GE feedback (including $P'(Y) < 0$ and the pricing feedback), stationary solutions (V, m) of the coupled HJB–FP system coincide with stationary solutions of* (I) *conditional on x, with $V(k, z, x; m) = U(k, z, x, m)$.*

extbfEquivalence and uniqueness. Establishing uniqueness in this GE setting requires generalizing the Lasry–Lions monotonicity condition to account for the pricing externalities. This depends on the properties of the equilibrium consumption process and how it responds to changes in the distribution m.

H Boundary and Regularity Conditions

(This section remains largely unchanged, emphasizing the reflecting boundary at $k = 0$, growth conditions on U, and integrability of m. The regularity of the pricing kernel is now crucial, as noted in **??**.)

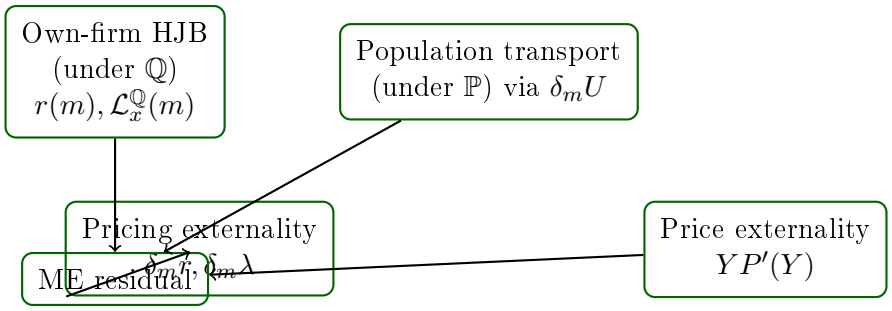

Figure 1: Schematic composition of the stationary master equation: HJB contributions, population transport, price externality, and the newly introduced pricing externalities.

I Computation: Methodologies and Implementation

Solving the GE mean-field system presents significant computational challenges because the state (k, z, x) is at least three-dimensional and interacts with the endogenous distribution m. Classical PDE solvers (Routes A and B) quickly encounter the curse of dimensionality, whereas probabilistic formulations (Route C) rely on Monte Carlo sampling and deep learning surrogates to remain tractable in higher dimensions.

Modern Computational Approaches. Deep-learning-based solvers have demonstrated substantial gains for high-dimensional PDEs and FBSDEs; see [**?**] and the probabilistic macro-finance applications in [**?**]. Route C adapts these insights to the present GE model, while Routes A and B draw on the numerical mean-field games literature.

I.1 Discretization for PDE Methods: FVM and Upwinding

Routes A and B require a stable discretization of the state space (k, z, x). The transport term in the k-direction lacks diffusion, so careless differencing leads to numerical oscillations.

HJB Discretization. The HJB equation (4) is typically solved via implicit finite differences on a tensor grid. The drift term $V_k(i^* - \delta k)$ mandates an upwind stencil: when $i^* - \delta k > 0$ the derivative uses left-biased differences, and vice versa.

FP Discretization and FVM. The stationary FP equation (5) is a conservation law for the mass of firms. Following [**?**], we discretize the k-dimension with the finite volume method (FVM) to guarantee conservation. Partition k into cells $[k_{j-1/2}, k_{j+1/2}]$ and define $m_j \approx \int_{k_{j-1/2}}^{k_{j+1/2}} m(k)\,\mathrm{d}k$. Integrating the FP equation over a cell yields

$$\partial_t m_j = -\frac{F_{j+1/2} - F_{j-1/2}}{\Delta k_j} + (\mathcal{L}_z^* m)_j, \tag{5}$$

where $F_{j+1/2}$ is the numerical flux at the interface $k_{j+1/2}$. An upwind flux respects the direction of the drift $v_{j+1/2} = i^*(k_{j+1/2}) - \delta k_{j+1/2}$:

$$F_{j+1/2} = \max(v_{j+1/2}, 0)\, m_j + \min(v_{j+1/2}, 0)\, m_{j+1}. \tag{6}$$

Why FVM? FVM preserves $\sum_j m_j = 1$ exactly, ensuring no artificial creation or loss of firms. Upwinding aligns the discretization with the physical direction of flow, removing the spurious oscillations that appear with centered differences.

Verification: FVM Upwind Flux Structure (??).

```
import sympy as sp
v = sp.symbols("v", real=True)
m_j, m_jp1 = sp.symbols("m_j m_jp1", real=True)
F_upwind = sp.Max(v, 0) * m_j + sp.Min(v, 0) * m_jp1
v_pos = sp.symbols("v_pos", positive=True)
v_neg = sp.symbols("v_neg", negative=True)
assert F_upwind.subs(v, v_pos) == v_pos * m_j
assert F_upwind.subs(v, v_neg) == v_neg * m_jp1
print("FVM upwind flux structure verified symbolically.")
```

I.2 Route A: HJB–FP Fixed Point (General Equilibrium)

The GE closure requires solving for the equilibrium distribution $m(x)$ simultaneously across all aggregate states x to determine the consumption dynamics and asset prices.

Algorithm A.1 (GE Fixed Point).

A.1 Discretize x. Define a grid for the aggregate state x.

A.2 Initialize $m^{(0)}(x)$. Initialize the distribution $m(x)$ for all x on the grid.

A.3 Aggregate Dynamics. Given $m^{(n)}(x)$, compute $C^{(n)}(x) = D(m^{(n)}(x), x)$. Estimate $\mu_C^{(n)}(x)$ and $\sigma_C^{(n)}(x)$ via numerical derivatives of $C^{(n)}(x)$.

A.4 Pricing. Compute $r^{(n)}(x), \lambda^{(n)}(x)$ using **??** and assemble $\mathcal{L}_x^{\mathbb{Q}(n)}$.

A.5 HJB step. Solve **??** for $V^{(n)}(x)$ on the tensor grid using the upwind discretization from **??**; recover $i^{*,(n)}(x)$.

A.6 FP step. Solve stationary **??** for $\widehat{m}^{(n+1)}(x)$ with the FVM scheme (**??**)–(**??**).

A.7 Update. Dampen and update $m^{(n+1)}(x)$. Iterate until convergence of $m(x), r(x), \lambda(x)$.

Computational Note: PDE Limitations. Route A is robust but suffers from the curse of dimensionality. Each additional state roughly multiplies the grid size, and diffusion terms require Hessians such as V_{xx}, which dominate runtime.

I.3 Route B: Direct Master-PDE Collocation (General Equilibrium)

We parameterize $U_\omega(k, z, x, \cdot)$ and $\delta_m U_\psi(\xi; k, z, x, \cdot)$. We must also parameterize the pricing functions $r_\theta(m), \lambda_\theta(m)$ and, crucially, their Lions derivatives $\delta_m r_\theta, \delta_m \lambda_\theta$.

Residual construction. At each collocation tuple $(k, z, x; \{\xi^n\})$, compute the ME residual $\widehat{\mathcal{R}}_{\text{ME}}$ (Appendix A), including the empirical estimate of the pricing externality:

$$\widehat{E}_{\text{pricing}} = \frac{1}{N}\sum_{n=1}^{N}\left[-(\delta_m r_\theta(\xi^n))U_\omega + (-\sigma_x \delta_m \lambda_\theta(\xi^n))(U_\omega)_x\right].$$

Minimize the empirical mean of $\widehat{\mathcal{R}}^2_{\text{ME}}$ plus penalties enforcing consistency between r_θ, λ_θ and the SDF derived from the implied consumption dynamics.

Complexity of Route B in GE. Computing and integrating the Lions derivatives $\delta_m r$ and $\delta_m \lambda$ is the main bottleneck. These derivatives inherit the full general equilibrium feedback of $C(x, m)$, so collocation remains practical only in low dimensions.

I.4 Route C: Probabilistic Approach (Deep BSDE)

A scalable alternative solves the HJB equation through its probabilistic representation as a backward SDE, leveraging neural networks to approximate the value function and its gradients [**?**].

Probabilistic Formulation (FBSDE). Let $V_t = V(k_t, z_t, x_t; m_t)$. Under the risk-neutral measure $\mathbb{Q}$, the martingale representation yields

$$\mathrm{d}V_t = \left(r_t V_t - \mathcal{H}^*_t\right)\mathrm{d}t + \Psi^z_t\,\mathrm{d}W_t + \Psi^x_t\,\mathrm{d}B^{\mathbb{Q}}_t, \tag{7}$$

where $\mathcal{H}^*_t$ is the maximized Hamiltonian and Ψ^z_t, Ψ^x_t are volatility terms proportional to V_z and V_x. Together with the forward SDEs for (k_t, z_t, x_t) this forms an FBSDE system.

Advantage: Avoiding Hessians. Deep BSDE solvers replace direct computation of V_{xx} and V_{zz} with automatic differentiation of neural networks, sidestepping the main bottleneck in PDE methods while scaling gracefully with dimensionality.

Algorithm C.1: Deep BSDE (Forward Euler Scheme).

C.1 Initialize networks. Parameterize V_Θ, its gradients (e.g., $(V_k)_\Theta$), and volatility surrogates Ψ_Θ. Initialize the pricing functions $r(m)$ and $\lambda(m)$.

C.2 Simulate paths. Draw M Monte Carlo paths of shocks (W_t, B_t) with initial states sampled from m_0. Set $Y_0 = V_\Theta(X_0)$.

C.3 Forward step. For each time increment:

a. Evaluate policies i^*_t from $(V_k)_\Theta$ and obtain volatilities Ψ_t from Ψ_Θ.

b. Update states X_{t+1} via the forward SDEs under $\mathbb{Q}$.

c. Propagate Y_{t+1} using the Euler discretization of (**??**).

d. Compute the network prediction $\widehat{Y}_{t+1} = V_\Theta(X_{t+1})$.

C.4 Loss minimization. Minimize the pathwise Markov mismatch

$$\mathcal{L}(\Theta) = \frac{1}{M(T/\Delta t)}\sum_{m=1}^{M}\sum_{t}\left\|Y^{(m)}_t - \widehat{Y}^{(m)}_t\right\|^2.$$

C.5 Equilibrium update. Update m (from simulated paths) and recompute $r(m), \lambda(m)$. Iterate steps 2–4 until convergence.

High-Dimensional Controls.
citechen$_h$uang$_2$025$_c$ontrolscombine$DeepBSDEsolverswithFVM$ $-$
$stylediscretizationsformodelsfeaturinghigh-dimensionalcontrols, illustratinghowprobabilisticandPDEtoo$

J Verification and Diagnostics

Residual norms. For collocation tuples (k, z, x, m):

$$\begin{aligned}
\mathcal{R}_{\text{HJB}} &\equiv r(x,m)\,V - \max_i\{\pi + V_k\,(i-\delta k) + \mathcal{L}_z V + \mathcal{L}_x^{\mathbb{Q}} V\},\\
\mathcal{R}_{\text{FP}} &\equiv -\partial_k\big[(i^*-\delta k), m\big] + \mathcal{L}_z^* m,\\
\mathcal{R}_{\text{Pricing}} &\equiv \|r(x) - (\rho + \gamma\mu_C(x) - \cdots)\| + \|\lambda(x) - \gamma\sigma_C(x)\|,\\
\mathcal{R}_{\text{ME}} &\equiv r(x,m)\,U - \Big(\text{HJB terms} + \text{Transport} + E_{\text{price}} + E_{\text{pricing}}\Big).
\end{aligned}$$

Sanity checks.

- *Risk-neutral case.* If $\gamma = 0$, then $\lambda = 0$ and $r = \rho$. The pricing externalities vanish, and $\mathcal{L}_x^{\mathbb{Q}} = \mathcal{L}_x$. The model must collapse to the partial equilibrium version.
- *Risk aversion sweep.* Increasing γ raises the market price of risk λ (typically), reducing the risk-adjusted drift of x and affecting valuation V.

K Economics: Aggregation, Irreversibility, and Asset Pricing

Aggregation. Aggregation enters via the general equilibrium externalities. The mean-field formulation explicitly decomposes these into the product market feedback (E_{price}) and the financial market feedback (E_{pricing}).

Irreversibility and Asset Prices. The asymmetry $\phi_- > \phi_+$ (costly reversibility) interacts with the endogenous SDF to generate the value premium (Zhang, 2005). If the market price of risk $\lambda(x)$ is countercyclical, firms constrained by adjustment costs become riskier in recessions, leading to higher expected returns.

Comparative statics.

- Larger η (steeper demand) amplifies the negative price externality, reducing investment.
- Higher γ (risk aversion) increases the sensitivity of asset prices to consumption dynamics, amplifying the pricing externality and typically increasing risk premia.
- Higher σ_x increases aggregate risk. This raises λ (if $\gamma > 0$) and deepens precautionary effects, with ambiguous effects on average investment depending on the balance of risk adjustment and discount rate effects.

L Appendix B: Residual-Loss Template (for implementation)

For a collocation tuple (k, z, x), an empirical measure $m = \frac{1}{N}\sum_{n=1}^{N} \delta_{\xi^n}$, and parameterized $U_\omega, \delta_m U_\psi, r_\theta, \lambda_\theta$ and their Lions derivatives $\delta_m r_\theta, \delta_m \lambda_\theta$.

$$
\begin{aligned}
\widehat{Y} &\equiv \frac{1}{N}\sum_{n=1}^{N} e^{x+\zeta^n}(\kappa^n)^\alpha, \\
\widehat{\mu_x^{\mathbb{Q}}} &\equiv \mu_x(x) - \sigma_x \lambda_\theta(x, m), \\
\widehat{E}_{\text{pricing}} &\equiv \frac{1}{N}\sum_{n=1}^{N} \left[-(\delta_m r_\theta(\xi^n))U_\omega + (-\sigma_x \delta_m \lambda_\theta(\xi^n))(U_\omega)_x\right], \\
\widehat{\mathcal{R}}_{\text{ME}} &\equiv r_\theta(x, m)\, U_\omega \\
&\quad - \max_i \left\{\pi + (U_\omega)_k\,(i - \delta k) + \mathcal{L}_z U_\omega + \mathcal{L}_x^{\mathbb{Q}}(\lambda_\theta) U_\omega\right\} \\
&\quad - \text{Transport}(\delta_m U_\psi) \\
&\quad - e^{x+z} k^\alpha\, \widehat{Y}\, P'(\widehat{Y}) - \widehat{E}_{\text{pricing}}.
\end{aligned}
$$

Minimize the loss $\mathcal{L} = \mathbb{E}\big[\widehat{\mathcal{R}}_{\text{ME}}^2\big]$ plus penalties and consistency conditions ensuring r_θ, λ_θ match the SDF derived from the implied consumption dynamics.

M Appendix C: Common-Noise Master Equation (Reference Note)

When the population law m_t itself diffuses under common noise, the functional Itô calculus on $\mathcal{P}_2$ introduces a second-order term in the measure variable. The stationary master equation would add a term related to the covariance of the common noise and second-order measure derivatives of U. Because this paper conditions on x, these terms are absent.

N Appendix D: Tiny Pseudocode (Plain `listings`)

(The pseudocode structure remains similar to the original but must incorporate the endogenous pricing functions and the pricing externalities in the residual calculation, as illustrated in Appendix B.)
Example pseudocode for the Deep BSDE loss used in Route C:

```
def compute_deep_bsde_loss(params, X0, dW_Q, dt, m_equilibrium):
    # Forward Euler Deep BSDE loss enforcing the Markov property
    T_steps = dW_Q.shape[0]
    X = X0
    V_sim = V_net(X0, params)
    loss = 0.0

    # Price kernel under the current equilibrium measure
    r_eq, lambda_eq = compute_equilibrium_prices(m_equilibrium)

    for t in range(T_steps):
        # 1. Evaluate gradients and diffusion terms from the networks
        V_k, Psi = compute_gradients_and_Psi(V_net, X, params)
```

```
        # 2. Optimal policy and generator inputs
        i_star = compute_policy(V_k)
        r_t = interpolate(r_eq, X)
        pi_t = compute_pi(X, i_star)

        # 3. Forward Euler step for the BSDE value process
        drift_V = r_t * V_sim - pi_t
        V_sim = V_sim + drift_V * dt + (Psi * dW_Q[t]).sum(axis=-1)

        # 4. Simulate the forward state (risk-neutral dynamics)
        X = simulate_X_forward_Q(X, i_star, dt, dW_Q[t], lambda_eq)

        # 5. Markov property loss
        V_target = V_net(X, params)
        loss += ((V_sim - V_target) ** 2).mean()

    return loss / T_steps
```

O Appendix E: Symbolic Verification (SymPy, static)

This appendix shows minimal SymPy checks to verify key derivations used in the text. For portability in this repository, we render these as code listings using `minted`. To execute, run the snippet in a Python environment with SymPy installed.

```
import sympy as sp

# 1) Isoelastic simplification:  Y P'(Y) = -eta P(Y)
Y, eta = sp.symbols('Y eta', positive=True)
P = Y**(-eta)
check1 = sp.simplify(Y*sp.diff(P, Y) + eta*P)
assert check1 == 0
print("Isoelastic: Y*P'(Y) = -eta*P(Y)  [OK]")

# 2) KKT/FOC solution for i* with asymmetric quadratic costs
i, k, p, phi_plus, phi_minus = sp.symbols('i k p phi_plus phi_minus', positive=True)
h_plus  = 0.5*phi_plus*i**2/k
FOC_plus  = sp.Eq(sp.diff(-i - h_plus + p*i, i), 0)
sol_plus  = sp.solve(FOC_plus, i)[0]
h_minus = 0.5*phi_minus*i**2/k
FOC_minus = sp.Eq(sp.diff(-i - h_minus + p*i, i), 0)
sol_minus = sp.solve(FOC_minus, i)[0]
assert sp.simplify(sol_plus  - k*(p-1)/phi_plus)  == 0
assert sp.simplify(sol_minus - k*(p-1)/phi_minus) == 0
print('KKT/FOC piecewise i* formulas     [OK]')

# 3) Envelope property for Hamiltonian in p: dH/dp = i*(p)
H_plus  = (-i - h_plus + p*i).subs(i, sol_plus)
dHp_dp  = sp.simplify(sp.diff(H_plus, p))
assert sp.simplify(dHp_dp - sol_plus) == 0
print('Envelope: dH/dp equals i*(p)       [OK]')

# 4) SDF dynamics and pricing kernel components (Ito's lemma check)
# Verification of CCAPM short rate and market price of risk
t = sp.symbols('t', positive=True)
rho, gamma = sp.symbols('rho gamma', positive=True)
```

```
C = sp.Function('C')(t)
mu_C, sigma_C = sp.symbols('mu_C sigma_C', real=True)

# SDF: Lambda = f(t, C) = exp(-rho*t) * C**(-gamma)
# Assume dC = C*mu_C dt + C*sigma_C dB
f = sp.exp(-rho*t) * C**(-gamma)
df_dt = sp.diff(f, t)
df_dC = sp.diff(f, C)
df_dC2 = sp.diff(f, C, 2)

# Drift of dLambda (Ito's lemma)
drift_Lambda = df_dt + df_dC * (C*mu_C) + 0.5 * df_dC2 * (C*sigma_C)**2
# Drift of dLambda/Lambda = -r dt
drift_dLambda_Lambda = sp.simplify(drift_Lambda / f)

# Target risk-free rate r (Ramsey rule + precautionary savings)
r_target = rho + gamma*mu_C - 0.5*gamma*(gamma+1)*sigma_C**2
assert sp.simplify(drift_dLambda_Lambda + r_target) == 0

# Diffusion of dLambda
diff_Lambda = df_dC * (C*sigma_C)
# Diffusion of dLambda/Lambda = -lambda dB
lambda_actual = sp.simplify(-diff_Lambda / f)
lambda_target = gamma*sigma_C
assert sp.simplify(lambda_actual - lambda_target) == 0

print('SDF dynamics (r and lambda) via Ito [OK]')

print('\nAll SymPy verification checks passed.')
```

Vector Itô lemma check (explicit grad/Hessian)

```
import sympy as sp

# State: X = (x1, x2), with SDE dX = b(X) dt + Sigma(X) dW, W is 2D
x1, x2 = sp.symbols('x1 x2', real=True)
t = sp.symbols('t', real=True)
f = sp.Function('f')

# Choose a concrete scalar function f(x1,x2,t) to avoid purely abstract tensor caveats
g = sp.exp(t) * (x1**2 * x2 + sp.sin(x2))

# Drift and diffusion (2x1 Brownian vector): b in R^2, Sigma in R^{2x2}
b1 = x1 + x2
b2 = x1 - x2
Sigma = sp.Matrix([[x1, 0],
                   [0,  x2]])  # diagonal for clarity

# Gradients and Hessian
grad = sp.Matrix([sp.diff(g, x1), sp.diff(g, x2)])
H = sp.hessian(g, (x1, x2))

# Time derivative
g_t = sp.diff(g, t)

# Compute 0.5 * trace(Sigma Sigma^T H)
SS = Sigma * Sigma.T
ito_diff_term = sp.Rational(1,2) * (SS.multiply_elementwise(H.T)).sum()  # since H is
↪  symmetric, trace(SS*H) = sum(SS .* H)

# Ito drift per lemma: g_t + grad·b + 0.5 trace(SS H)
drift_ito = sp.simplify(g_t + (grad.T * sp.Matrix([b1, b2]))[0] + ito_diff_term)

# Build the explicit target by manual expansion to cross-check (equivalent expression)
# Target: e^t * [ (x1^2*x2 + sin x2) + (dg/dx1)*(x1+x2) + (dg/dx2)*(x1-x2) +
↪  0.5*(x1^2*g_x1x1 + x2^2*g_x2x2) ]
gx1 = sp.diff(g, x1)
gx2 = sp.diff(g, x2)
gxx11 = sp.diff(g, x1, 2)
gxx22 = sp.diff(g, x2, 2)
target = g_t + gx1*(x1 + x2) + gx2*(x1 - x2) + sp.Rational(1,2)*(x1**2*gxx11 + x2**2*gxx22)

assert sp.simplify(drift_ito - target) == 0
print("Vector Ito drift matches target [OK]")
```

Itô drift equals HJB operator under $\mathbb{Q}$

```
import sympy as sp

# States and function V(k,z,x)
k, z, x = sp.symbols('k z x', real=True)
V = sp.Function('V')(k, z, x)

# Parameters and policy
delta = sp.symbols('delta', positive=True)
i_star = sp.Function('i_star')(k, z, x)  # optimal policy (kept symbolic)

# Risk-neutral x-drift and diffusions
mu_xQ, sigma_x, sigma_z = sp.symbols('mu_xQ sigma_x sigma_z', real=True)
mu_z = sp.Function('mu_z')(z)

# Generators Lz and Lx^Q (acting on V)
Lz_V = mu_z * sp.diff(V, z) + (sp.Rational(1,2) * sigma_z**2) * sp.diff(V, z, 2)
LxQ_V = mu_xQ * sp.diff(V, x) + sp.Rational(1,2) * sigma_x**2 * sp.diff(V, x, 2)

# Capital drift term
drift_k_term = sp.diff(V, k) * (i_star - delta * k)

# Itô drift of V under Q (no k-diffusion; z and x diffusions as above)
ito_drift_V = drift_k_term + Lz_V + LxQ_V

# HJB RHS terms (without max, taking i_star)
pi = sp.Function('pi')(k, i_star, z, x)
HJB_rhs = pi + drift_k_term + Lz_V + LxQ_V

# Check that the constructed Itô drift equals the operator part of HJB_rhs
assert sp.simplify(HJB_rhs - pi - ito_drift_V) == 0
print("Itô drift matches HJB operator terms [OK]")
```

Risk-neutral drift transformation consistency

```
import sympy as sp

mu_x, sigma_x, gamma = sp.symbols('mu_x sigma_x gamma', real=True)
x = sp.symbols('x', real=True)
C = sp.Function('C')(x)

# sigma_C = (C'/C) sigma_x, lambda = gamma * sigma_C
sigma_C_expr = sp.diff(C, x) / C * sigma_x
lambda_expr = gamma * sigma_C_expr

# Q-drift: mu_x^Q = mu_x - sigma_x * lambda
mu_xQ_expr = mu_x - sigma_x * lambda_expr

# Expected symbolic structure
target = mu_x - gamma * (sp.diff(C, x) / C) * sigma_x**2
assert sp.simplify(mu_xQ_expr - target) == 0
print("Risk-neutral drift transformation (mu_x -> mu_x^Q) consistent [OK]")
```

Interactive Itô and operator checks (pycon transcript)

```
>>> import sympy as sp
>>> x1, x2, t = sp.symbols('x1 x2 t', real=True)
>>> g = sp.exp(t) * (x1**2 * x2 + sp.sin(x2))
>>> b1, b2 = x1 + x2, x1 - x2
>>> Sigma = sp.Matrix([[x1, 0],[0, x2]])
>>> grad = sp.Matrix([sp.diff(g, x1), sp.diff(g, x2)])
>>> H = sp.hessian(g, (x1, x2))
>>> g_t = sp.diff(g, t)
>>> SS = Sigma*Sigma.T
>>> ito_diff = sp.Rational(1,2) * (SS.multiply_elementwise(H.T)).sum()
>>> drift_ito = sp.simplify(g_t + (grad.T*sp.Matrix([b1,b2]))[0] + ito_diff)
>>> gx1, gx2 = sp.diff(g,x1), sp.diff(g,x2)
>>> gxx11, gxx22 = sp.diff(g,x1,2), sp.diff(g,x2,2)
>>> target = g_t + gx1*(x1+x2) + gx2*(x1-x2) + sp.Rational(1,2)*(x1**2*gxx11 + x2**2*gxx22)
>>> sp.simplify(drift_ito - target)
0
>>> # HJB operator check under Q for V(k,z,x)
>>> k, z, x = sp.symbols('k z x', real=True)
>>> V = sp.Function('V')(k, z, x)
>>> delta = sp.symbols('delta', positive=True)
>>> i_star = sp.Function('i_star')(k, z, x)
>>> mu_xQ, sigma_x, sigma_z = sp.symbols('mu_xQ sigma_x sigma_z', real=True)
>>> mu_z = sp.Function('mu_z')(z)
>>> Lz_V = mu_z*sp.diff(V,z) + sp.Rational(1,2)*sigma_z**2*sp.diff(V,z,2)
>>> LxQ_V = mu_xQ*sp.diff(V,x) + sp.Rational(1,2)*sigma_x**2*sp.diff(V,x,2)
>>> drift_k = sp.diff(V,k)*(i_star - delta*k)
>>> ito_drift = sp.simplify(drift_k + Lz_V + LxQ_V)
>>> pi = sp.Function('pi')(k, i_star, z, x)
>>> HJB_rhs = sp.simplify(pi + drift_k + Lz_V + LxQ_V)
>>> sp.simplify(HJB_rhs - pi - ito_drift)
0
```

References

[1] Bensoussan, A., J. Frehse, and P. Yam (2013). *Mean Field Games and Mean Field Type Control Theory.* Springer.

[2] Chen, H. and J. Huang (2025). Applications of Deep Learning-Based Probabilistic Approach to Economic Models with High-Dimensional Controls. *Working Paper*, The Chinese University of Hong Kong.

[3] Han, J., A. Jentzen, and W. E (2018). Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences* 115(34): 8505–8510.

[4] Huang, J. (2025). A Probabilistic Solution to High-Dimensional Continuous-Time Macro and Finance Models. *Working Paper*, The Chinese University of Hong Kong (CESifo Working Paper No. 10600).

[5] Carmona, R. and F. Delarue (2018). *Probabilistic Theory of Mean Field Games with Applications.* Springer.

[6] Cardaliaguet, P., F. Delarue, J.-M. Lasry, and P.-L. Lions (2019). *The Master Equation and the Convergence Problem in Mean Field Games.* Princeton University Press.

[7] Duffie, D. (2001). *Dynamic Asset Pricing Theory.* Princeton University Press.

[8] Lasry, J.-M. and P.-L. Lions (2007). Mean field games. *Japanese Journal of Mathematics* 2(1): 229–260.

[9] Zhang, L. (2005). The value premium. *Journal of Finance* 60(1): 67–103.