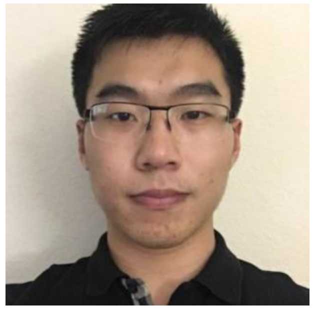
Deep Learning Asset Pricing

Oualid Missaoui

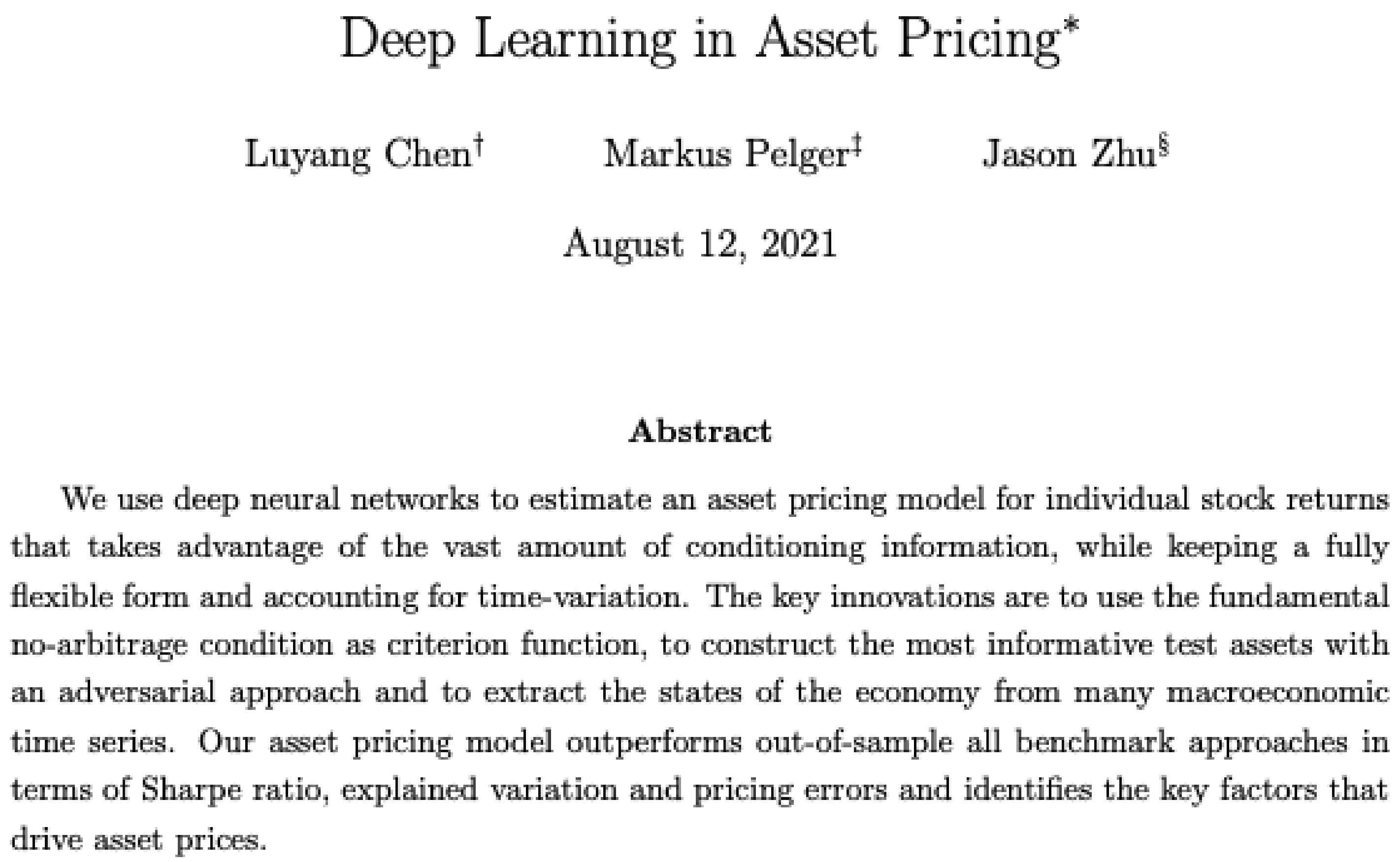
Agenda

Linkedin

Stanford



Stanford



Deep Learning in Asset Pricing*

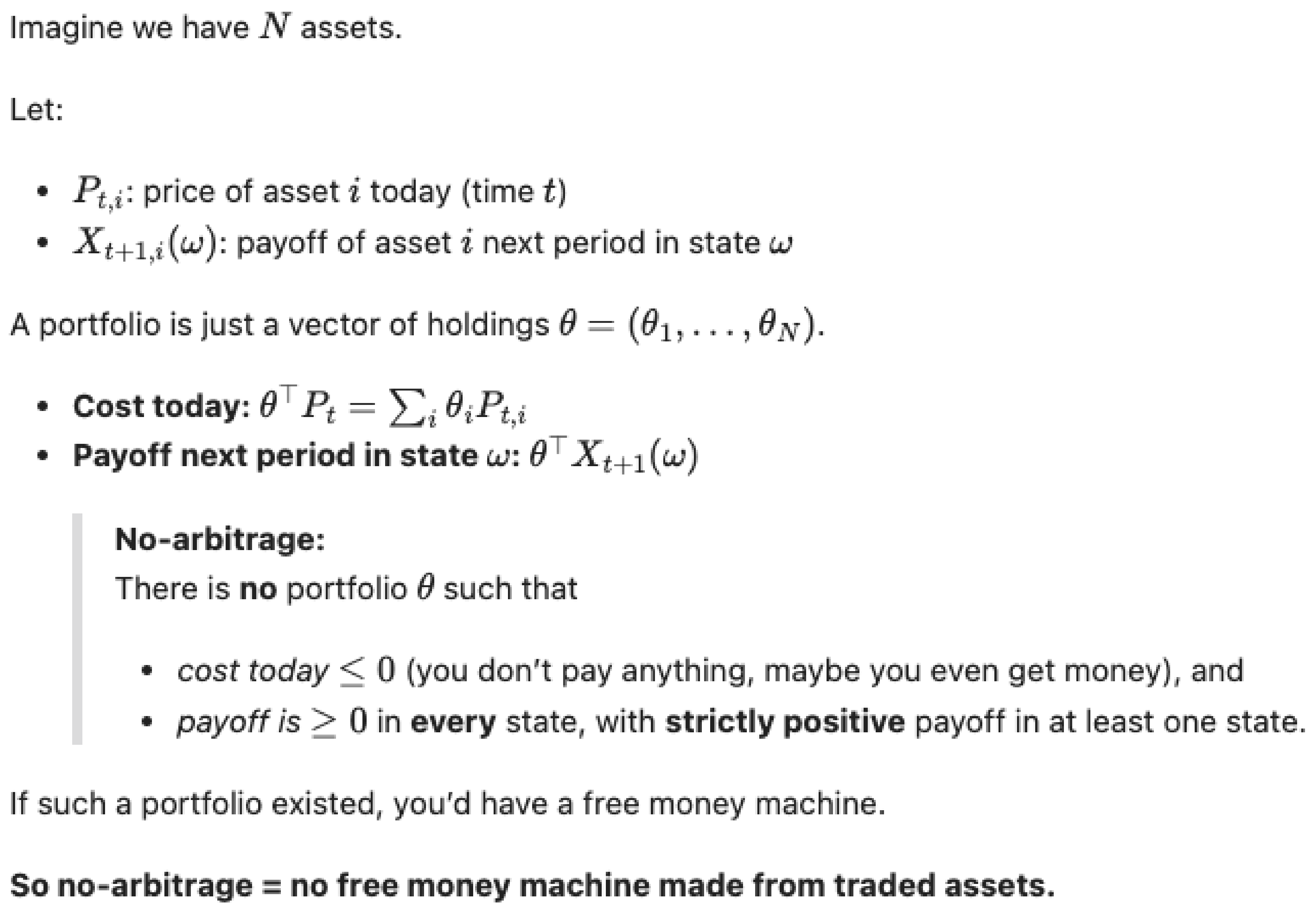
Luyang Chen† Markus Pelger‡ Jason Zhu§

August 12, 2021

Abstract

We use deep neural networks to estimate an asset pricing model for individual stock returns that takes advantage of the vast amount of conditioning information, while keeping a fully flexible form and accounting for time-variation. The key innovations are to use the fundamental no-arbitrage condition as criterion function, to construct the most informative test assets with an adversarial approach and to extract the states of the economy from many macroeconomic time series. Our asset pricing model outperforms out-of-sample all benchmark approaches in terms of Sharpe ratio, explained variation and pricing errors and identifies the key factors that drive asset prices.

What does no-arbitrage mean ?



Imagine we have N assets.

Let:

- $P_{t,i}$: price of asset i today (time t)
- $X_{t+1,i}(\omega)$: payoff of asset i next period in state ω

A portfolio is just a vector of holdings $\theta = (\theta_1, \ldots, \theta_N)$.

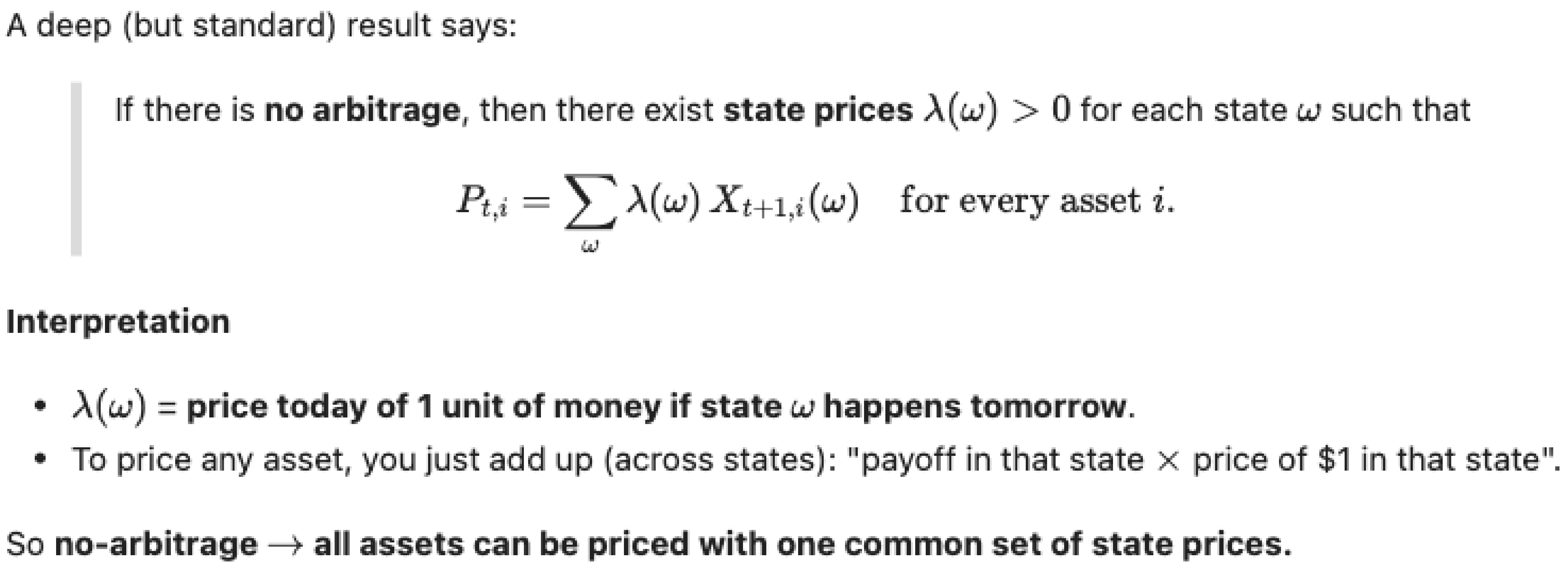
- **Cost today:** $\theta^\top P_t = \sum_i \theta_i P_{t,i}$
- **Payoff next period in state** ω**:** $\theta^\top X_{t+1}(\omega)$

> **No-arbitrage:**
> There is **no** portfolio θ such that
>
> - *cost today* ≤ 0 (you don't pay anything, maybe you even get money), and
> - *payoff is* ≥ 0 in **every** state, with **strictly positive** payoff in at least one state.

If such a portfolio existed, you'd have a free money machine.

So no-arbitrage = no free money machine made from traded assets.

From no-arbitrage to state prices



A deep (but standard) result says:

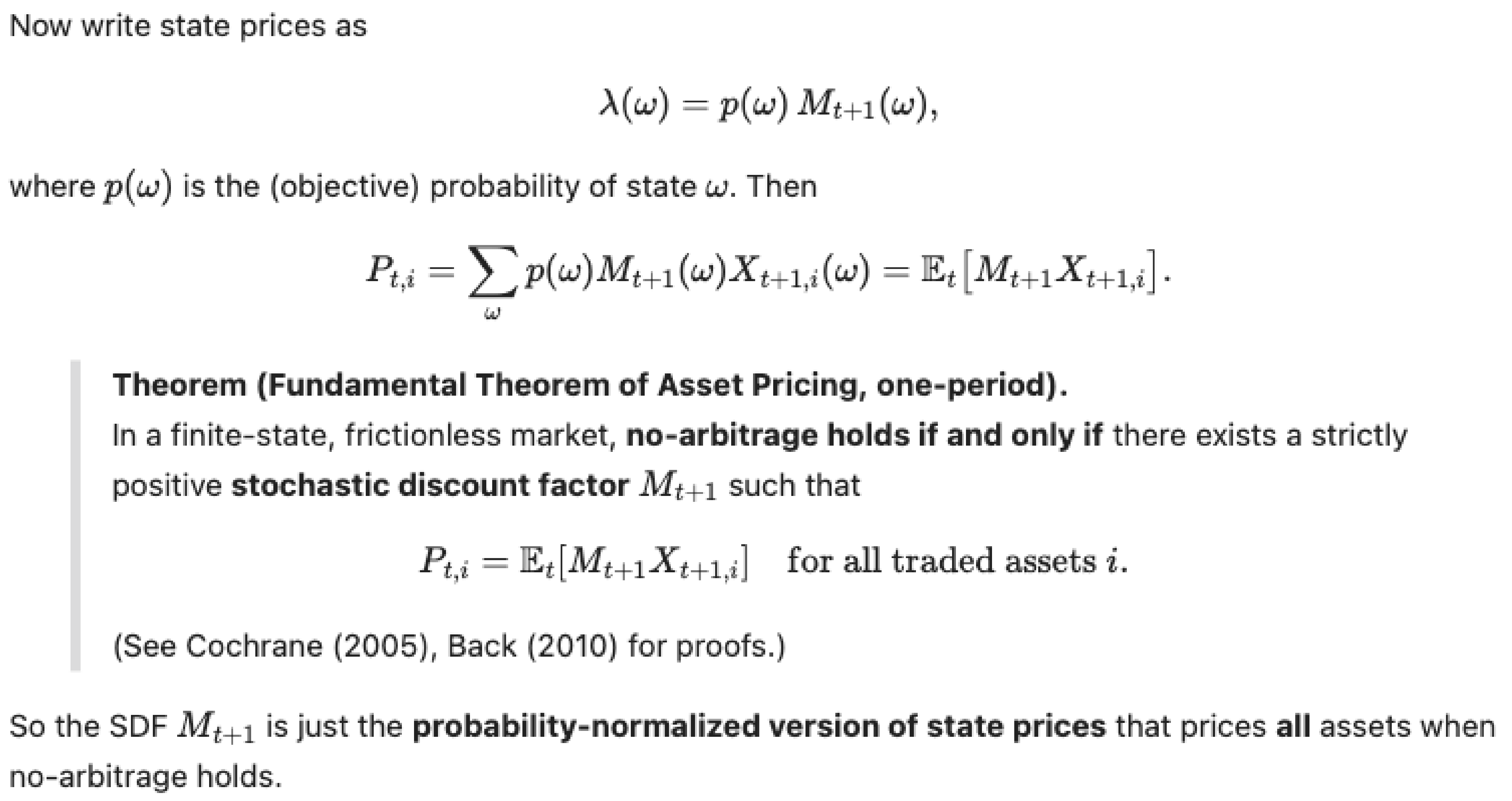
> If there is **no arbitrage**, then there exist **state prices** $\lambda(\omega) > 0$ for each state ω such that
>
> $$P_{t,i} = \sum_{\omega} \lambda(\omega)\, X_{t+1,i}(\omega) \quad \text{for every asset } i.$$

Interpretation

- $\lambda(\omega)$ = **price today of 1 unit of money if state ω happens tomorrow**.
- To price any asset, you just add up (across states): "payoff in that state $\times$ price of $1 in that state".

So **no-arbitrage $\rightarrow$ all assets can be priced with one common set of state prices.**

From state prices to the *stochastic discount factor* (SDF)



Now write state prices as

$$\lambda(\omega) = p(\omega)\, M_{t+1}(\omega),$$

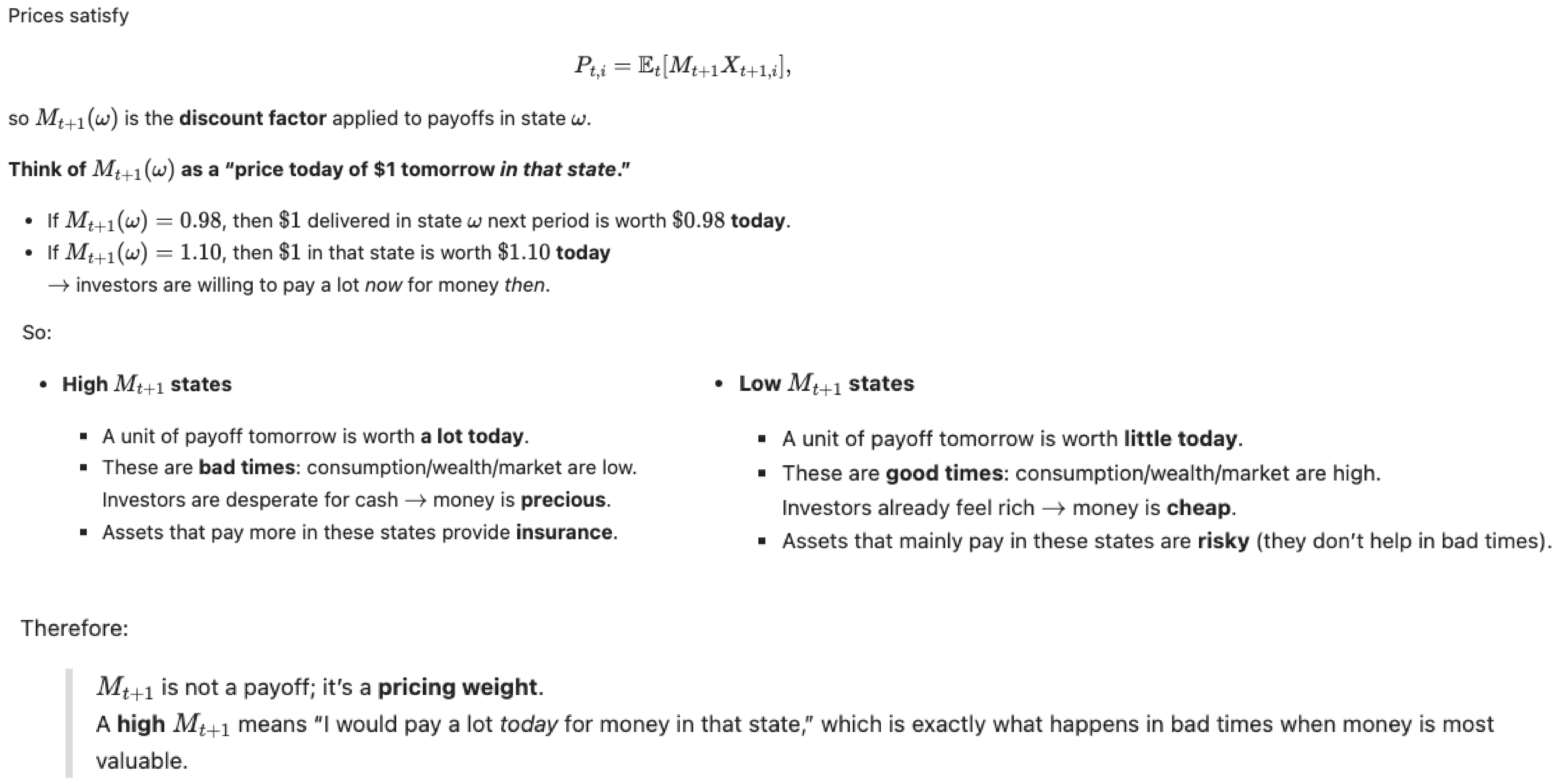
where $p(\omega)$ is the (objective) probability of state ω. Then

$$P_{t,i} = \sum_{\omega} p(\omega) M_{t+1}(\omega) X_{t+1,i}(\omega) = \mathbb{E}_t\left[M_{t+1} X_{t+1,i}\right].$$

> **Theorem (Fundamental Theorem of Asset Pricing, one-period).**
> In a finite-state, frictionless market, **no-arbitrage holds if and only if** there exists a strictly positive **stochastic discount factor** M_{t+1} such that
>
> $$P_{t,i} = \mathbb{E}_t[M_{t+1} X_{t+1,i}] \quad \text{for all traded assets } i.$$
>
> (See Cochrane (2005), Back (2010) for proofs.)

So the SDF M_{t+1} is just the **probability-normalized version of state prices** that prices **all** assets when no-arbitrage holds.

Intuition of the SDF



Prices satisfy

$$P_{t,i} = \mathbb{E}_t[M_{t+1}X_{t+1,i}],$$

so $M_{t+1}(\omega)$ is the **discount factor** applied to payoffs in state ω.

Think of $M_{t+1}(\omega)$ as a "price today of \$1 tomorrow *in that state*."

- If $M_{t+1}(\omega) = 0.98$, then \$1 delivered in state ω next period is worth $\$0.98$ **today**.
- If $M_{t+1}(\omega) = 1.10$, then \$1 in that state is worth $\$1.10$ **today**
  $\rightarrow$ investors are willing to pay a lot *now* for money *then*.

So:

- **High M_{t+1} states**
  - A unit of payoff tomorrow is worth **a lot today**.
  - These are **bad times**: consumption/wealth/market are low.
    Investors are desperate for cash $\rightarrow$ money is **precious**.
  - Assets that pay more in these states provide **insurance**.
- **Low M_{t+1} states**
  - A unit of payoff tomorrow is worth **little today**.
  - These are **good times**: consumption/wealth/market are high.
    Investors already feel rich $\rightarrow$ money is **cheap**.
  - Assets that mainly pay in these states are **risky** (they don't help in bad times).

Therefore:

> M_{t+1} is not a payoff; it's a **pricing weight**.
> A **high** M_{t+1} means "I would pay a lot *today* for money in that state," which is exactly what happens in bad times when money is most valuable.

Bring in the risk-free asset & the excess return condition

Suppose there is a **risk-free asset** with gross return R^f_{t+1}.

- Its payoff is $X^f_{t+1} = R^f_{t+1}$ for sure.
- Its price is 1 (invest one dollar today).

Price equation for the risk-free asset:

$$1 = \mathbb{E}_t[M_{t+1}R^f_{t+1}].$$

Now take any **risky asset** with gross return $R_{t+1,i}$ and excess return

$$R^e_{t+1,i} = R_{t+1,i} - R^f_{t+1}.$$

Pricing equation for that asset:

$$P_{t,i} = \mathbb{E}_t[M_{t+1}R_{t+1,i}].$$

If we express everything in "invest $1 today" units, this becomes:

$$1 = \mathbb{E}_t[M_{t+1}R_{t+1,i}].$$

Subtract the risk-free equation:

$$0 = \mathbb{E}_t[M_{t+1}R_{t+1,i}] - \mathbb{E}_t[M_{t+1}R^f_{t+1}] = \mathbb{E}_t\big[M_{t+1}(R_{t+1,i} - R^f_{t+1})\big].$$

So we get the key condition used in the paper:

$$\boxed{\mathbb{E}_t[M_{t+1}R^e_{t+1,i}] = 0 \quad \text{for all } i.}$$

This is the **"first SDF equation"** they write down.

From SDF to risk premia: co-movement with M

Start from the SDF pricing condition for excess returns:

$$0 = \mathbb{E}_t[M_{t+1} R^e_{t+1,i}].$$

Decompose into mean + covariance:

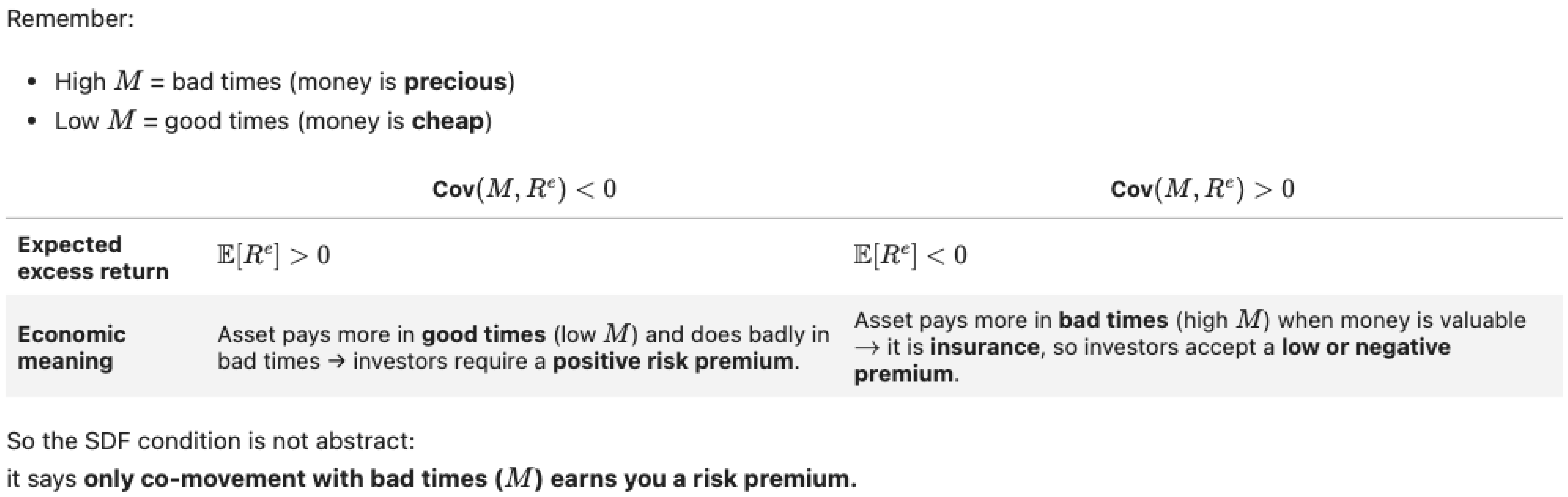
$$\mathbb{E}_t[M_{t+1} R^e_{t+1,i}] = \mathbb{E}_t[M_{t+1}]\, \mathbb{E}_t[R^e_{t+1,i}] + \mathrm{Cov}_t(M_{t+1}, R^e_{t+1,i}).$$

Solve for the expected excess return:

$$\boxed{\mathbb{E}_t[R^e_{t+1,i}] = -\frac{\mathrm{Cov}_t(M_{t+1}, R^e_{t+1,i})}{\mathbb{E}_t[M_{t+1}]}}$$

Since $\mathbb{E}_t[M_{t+1}] > 0$, the **sign** of $\mathbb{E}[R^e]$ is the **opposite** of the sign of $\mathrm{Cov}(M, R^e)$.

2x2 Interpretation: How co-movement with M affects risk premia



Remember:

- High M = bad times (money is **precious**)
- Low M = good times (money is **cheap**)

	Cov$(M, R^e) < 0$	**Cov**$(M, R^e) > 0$
Expected excess return	$\mathbb{E}[R^e] > 0$	$\mathbb{E}[R^e] < 0$
Economic meaning	Asset pays more in **good times** (low M) and does badly in bad times → investors require a **positive risk premium**.	Asset pays more in **bad times** (high M) when money is valuable $\rightarrow$ it is **insurance**, so investors accept a **low or negative premium**.

So the SDF condition is not abstract:

it says **only co-movement with bad times (M) earns you a risk premium.**

From SDF to portfolio weights

We start from the SDF pricing condition in excess returns:

$$\mathbb{E}_t\left[M_{t+1}R^e_{t+1,i}\right] = 0 \quad \text{for every asset } i.$$

Assume a **linear SDF in returns**:

$$M_{t+1} = 1 - \omega_t^\top R^e_{t+1},$$

where R^e_{t+1} is the $N \times 1$ vector of excess returns and ω_t is an $N \times 1$ vector of portfolio weights.

Plug this into the condition (in vector form):

$$\mathbb{E}_t\left[(1 - \omega_t^\top R^e_{t+1})R^e_{t+1}\right] = 0_{N\times 1}.$$

Expand the expectation:

$$\mathbb{E}_t[R^e_{t+1}] - \mathbb{E}_t\left[R^e_{t+1}R^{e\top}_{t+1}\right]\omega_t = 0.$$

Define

$$\mu_t = \mathbb{E}_t[R^e_{t+1}], \qquad \Sigma_t = \mathbb{E}_t[R^e_{t+1}R^{e\top}_{t+1}],$$

then the equation becomes

$$\mu_t - \Sigma_t\omega_t = 0 \quad \Rightarrow \quad \boxed{\omega_t = \Sigma_t^{-1}\mu_t}.$$

So **no-arbitrage + linear SDF** $\rightarrow$ **the SDF weights are** $\Sigma_t^{-1}\mu_t$.

Relation to the tangency portfolio

In mean–variance portfolio theory (with a risk-free asset), we look for the **risky portfolio with the highest Sharpe ratio**:

- Let $\mu_t = \mathbb{E}_t[R^e_{t+1}]$ (expected excess returns).
- Let $\Sigma_t = \mathrm{Var}_t(R^e_{t+1})$ (covariance matrix).

Sharpe ratio of a portfolio with weights ω:

$$\mathrm{Sharpe}(\omega) = \frac{\omega^\top \mu_t}{\sqrt{\omega^\top \Sigma_t \omega}}.$$

Because the Sharpe ratio is scale-invariant, we can fix the denominator and maximize the numerator. A convenient problem is:

$$\max_\omega \ \omega^\top \mu_t \quad \text{s.t.} \quad \omega^\top \Sigma_t \omega = 1.$$

Set up the Lagrangian:

$$\mathcal{L}(\omega, \lambda) = \omega^\top \mu_t - \lambda(\omega^\top \Sigma_t \omega - 1).$$

First-order condition w.r.t. ω:

$$\nabla_\omega \mathcal{L} = \mu_t - 2\lambda \Sigma_t \omega = 0 \quad \Rightarrow \quad \Sigma_t \omega = \frac{1}{2\lambda}\mu_t.$$

Therefore

$$\omega = \frac{1}{2\lambda}\Sigma_t^{-1}\mu_t \quad \Rightarrow \quad \boxed{\omega^* \propto \Sigma_t^{-1}\mu_t}.$$

So the **maximum-Sharpe (tangency) portfolio** has weights proportional to $\Sigma_t^{-1}\mu_t$.

Combining with the previous slide:

- The linear SDF is $M_{t+1} = 1 - \omega_t^\top R^e_{t+1}$.
- No-arbitrage implies $\omega_t = \Sigma_t^{-1}\mu_t$.
- But $\Sigma_t^{-1}\mu_t$ are exactly the **tangency portfolio weights**.

Conclusion: in the linear case, the SDF is an affine function of the return on the tangency (max-Sharpe) portfolio.

Why the “span of returns” SDF is general enough

Key idea: For pricing a given set of returns, we can always replace any SDF by one that is **linear in those returns** without changing any prices.

1. Start from any SDF

Let M_{t+1} be any stochastic discount factor that prices our N assets:

$$P_{t,i} = \mathbb{E}_t[M_{t+1}X_{t+1,i}], \quad i = 1, \dots, N.$$

Let R^e_{t+1} be the $N \times 1$ vector of excess returns.

2. Projection trick (Hansen–Jagannathan / Cochrane / Back)

We work in the Hilbert space of square–integrable random variables and **project** M_{t+1} onto the linear span of R^e_{t+1}:

$$M_{t+1} = \underbrace{\left(a_t - \omega_t^\top R^e_{t+1}\right)}_{\text{in span of returns}} + \underbrace{M^\perp_{t+1}}_{\text{orthogonal to all } R^e}.$$

By construction of the projection:

$$\mathbb{E}_t\left[M^\perp_{t+1} R^e_{t+1,i}\right] = 0 \quad \text{for every asset } i.$$

Hence, for each asset i:

$$\mathbb{E}_t[M_{t+1}R^e_{t+1,i}] = \mathbb{E}_t\left[(a_t - \omega_t^\top R^e_{t+1})R^e_{t+1,i}\right] + \underbrace{\mathbb{E}_t[M^\perp_{t+1}R^e_{t+1,i}]}_{0}.$$

So the **projected SDF**

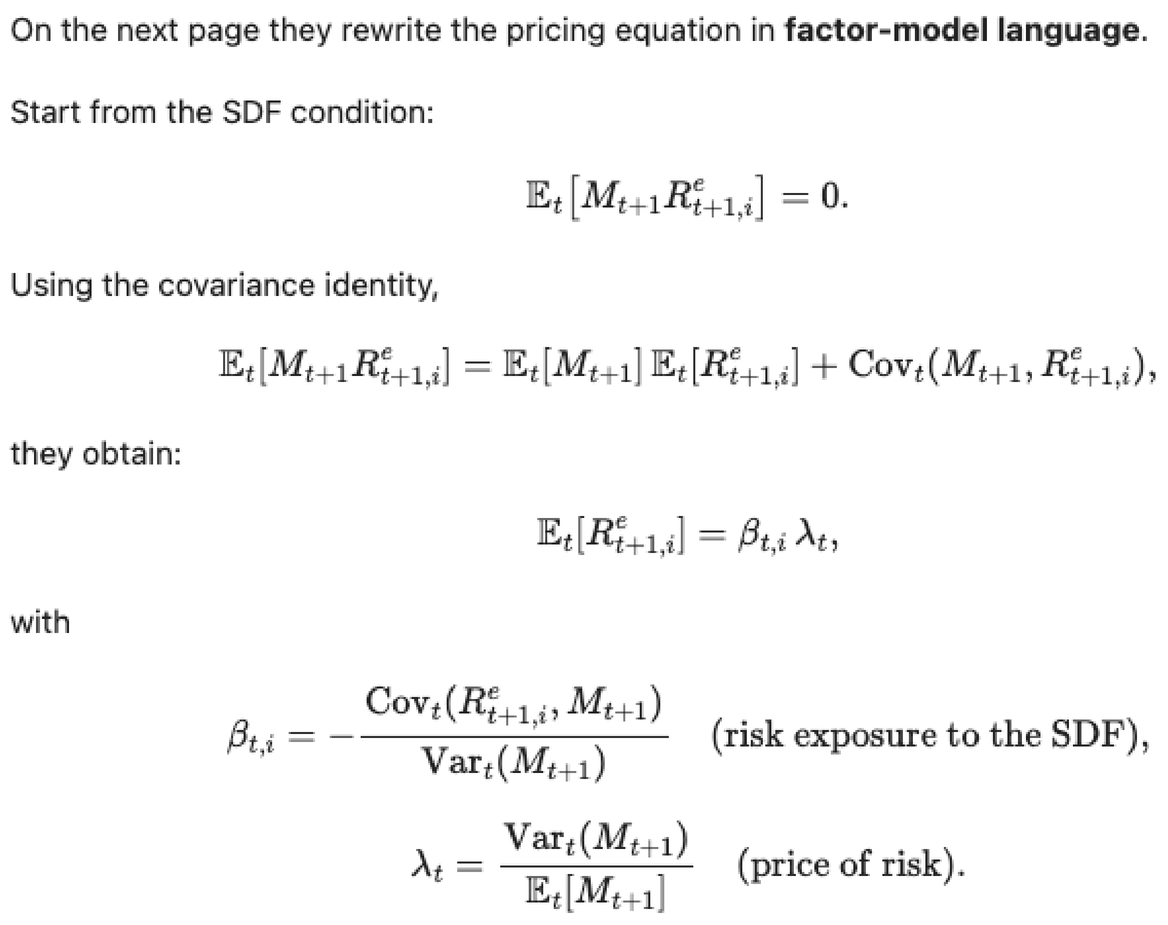
$$M^*_{t+1} = a_t - \omega_t^\top R^e_{t+1}$$

prices all assets exactly the same as the original M_{t+1}.

Because SDFs are defined up to a positive scale, we can normalize $a_t = 1$ and write

$$M^*_{t+1} = 1 - \omega_t^\top R^e_{t+1}.$$

Turn SDF / tangency portfolio into a one-factor model



On the next page they rewrite the pricing equation in **factor-model language**.

Start from the SDF condition:

$$\mathbb{E}_t\left[M_{t+1}R^e_{t+1,i}\right] = 0.$$

Using the covariance identity,

$$\mathbb{E}_t[M_{t+1}R^e_{t+1,i}] = \mathbb{E}_t[M_{t+1}]\,\mathbb{E}_t[R^e_{t+1,i}] + \mathrm{Cov}_t(M_{t+1}, R^e_{t+1,i}),$$

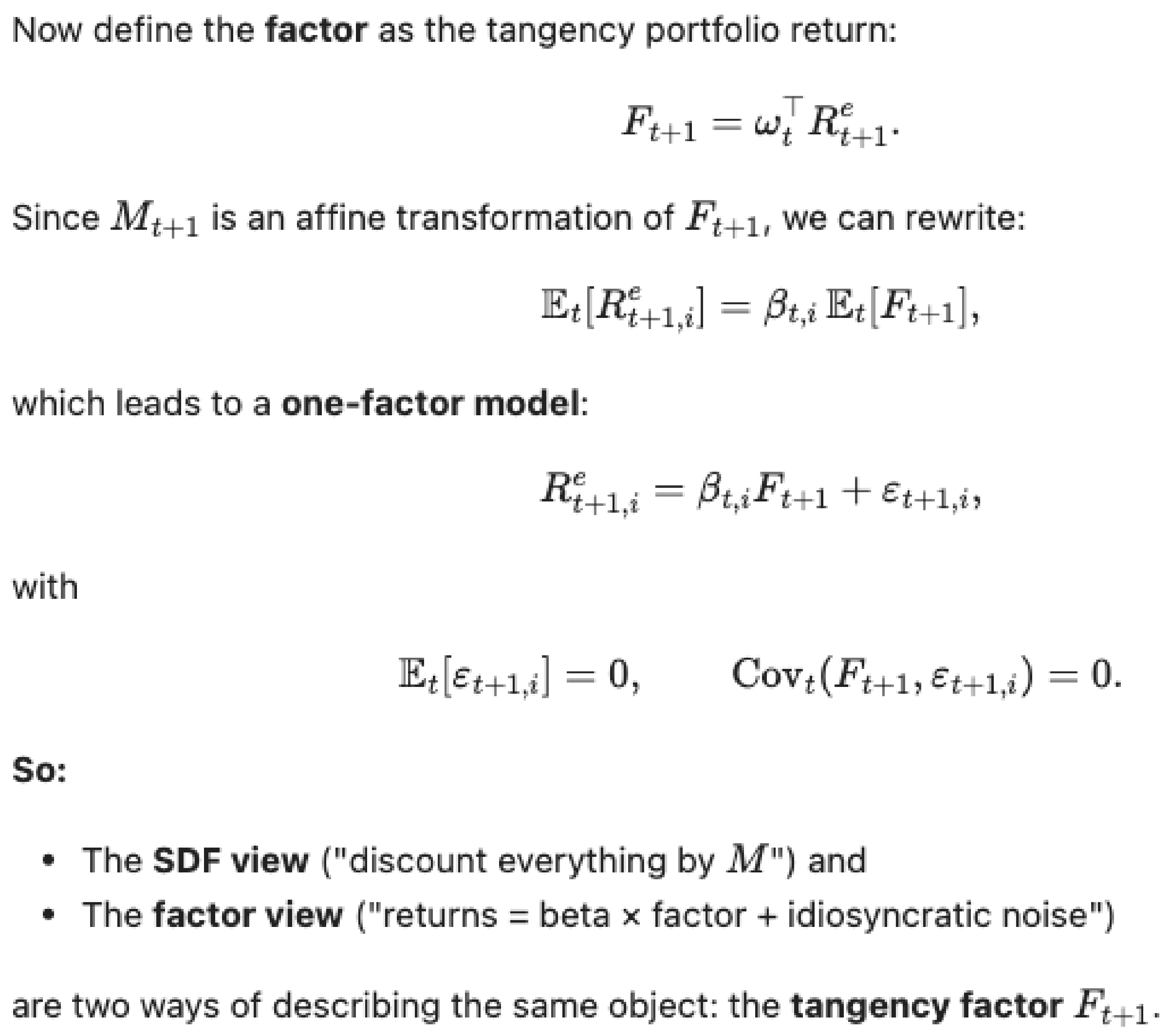
they obtain:

$$\mathbb{E}_t[R^e_{t+1,i}] = \beta_{t,i}\,\lambda_t,$$

with

$$\beta_{t,i} = -\frac{\mathrm{Cov}_t(R^e_{t+1,i}, M_{t+1})}{\mathrm{Var}_t(M_{t+1})} \quad \text{(risk exposure to the SDF)},$$

$$\lambda_t = \frac{\mathrm{Var}_t(M_{t+1})}{\mathbb{E}_t[M_{t+1}]} \quad \text{(price of risk)}.$$



Now define the **factor** as the tangency portfolio return:

$$F_{t+1} = \omega_t^\top R^e_{t+1}.$$

Since M_{t+1} is an affine transformation of F_{t+1}, we can rewrite:

$$\mathbb{E}_t[R^e_{t+1,i}] = \beta_{t,i}\,\mathbb{E}_t[F_{t+1}],$$

which leads to a **one-factor model**:

$$R^e_{t+1,i} = \beta_{t,i}F_{t+1} + \varepsilon_{t+1,i},$$

with

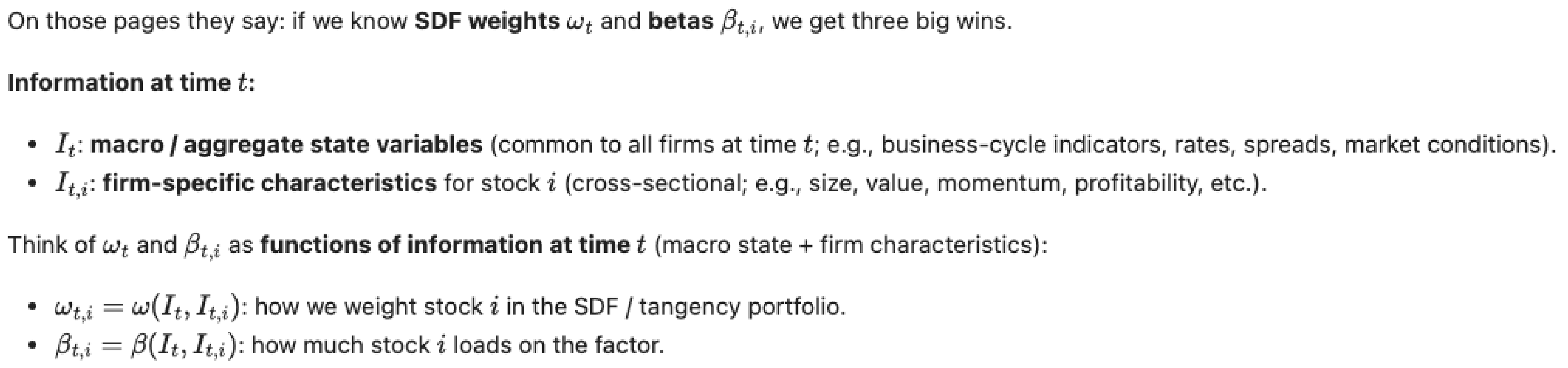
$$\mathbb{E}_t[\varepsilon_{t+1,i}] = 0, \qquad \mathrm{Cov}_t(F_{t+1}, \varepsilon_{t+1,i}) = 0.$$

So:

- The **SDF view** ("discount everything by M") and
- The **factor view** ("returns = beta × factor + idiosyncratic noise")

are two ways of describing the same object: the **tangency factor** F_{t+1}.

From SDF weights to economics: pricing, portfolios, decomposition



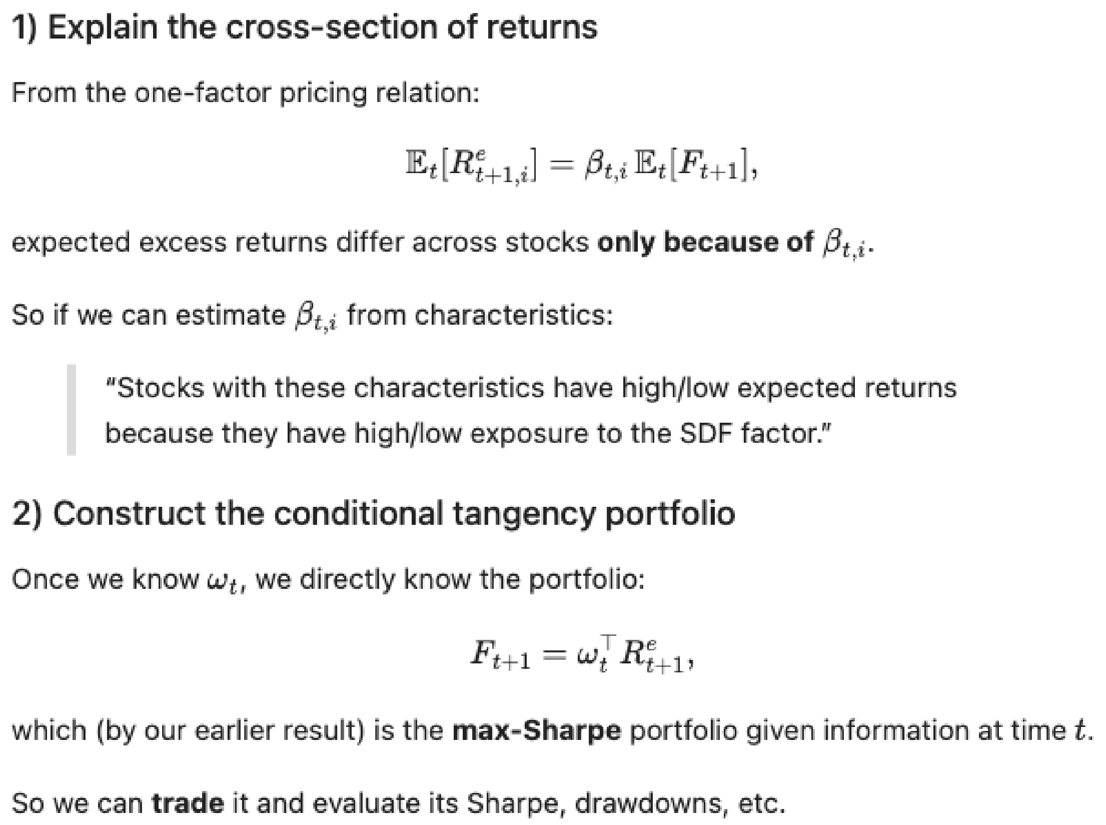
On those pages they say: if we know **SDF weights** ω_t and **betas** $\beta_{t,i}$, we get three big wins.

Information at time t:

- I_t: **macro / aggregate state variables** (common to all firms at time t; e.g., business-cycle indicators, rates, spreads, market conditions).
- $I_{t,i}$: **firm-specific characteristics** for stock i (cross-sectional; e.g., size, value, momentum, profitability, etc.).

Think of ω_t and $\beta_{t,i}$ as **functions of information at time t** (macro state + firm characteristics):

- $\omega_{t,i} = \omega(I_t, I_{t,i})$: how we weight stock i in the SDF / tangency portfolio.
- $\beta_{t,i} = \beta(I_t, I_{t,i})$: how much stock i loads on the factor.



1) Explain the cross-section of returns

From the one-factor pricing relation:

$$\mathbb{E}_t[R^e_{t+1,i}] = \beta_{t,i}\,\mathbb{E}_t[F_{t+1}],$$

expected excess returns differ across stocks **only because of $\beta_{t,i}$**.

So if we can estimate $\beta_{t,i}$ from characteristics:

> "Stocks with these characteristics have high/low expected returns because they have high/low exposure to the SDF factor."

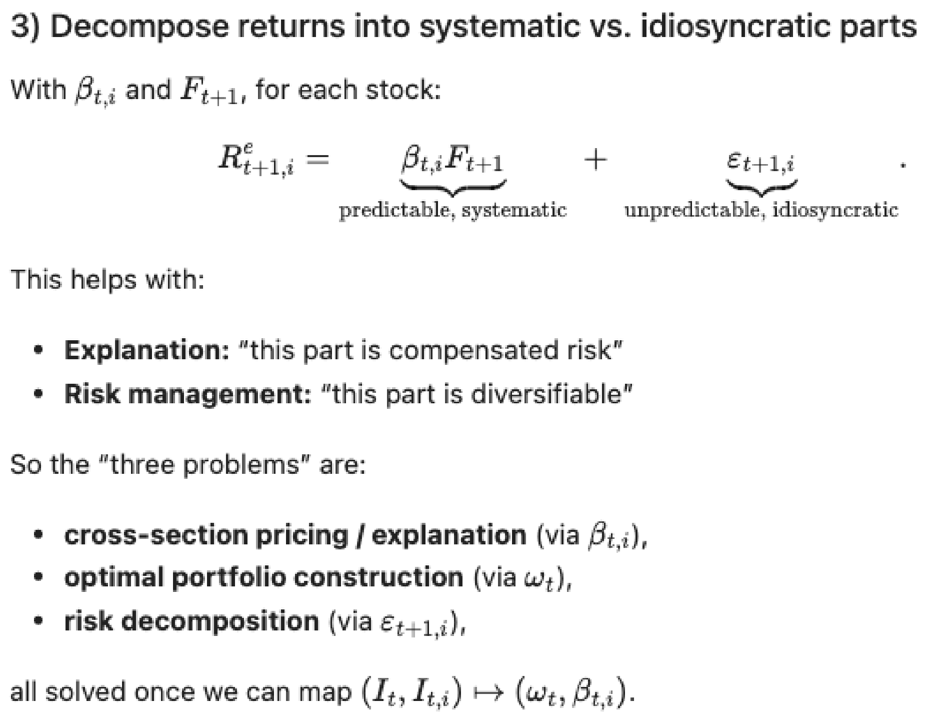
2) Construct the conditional tangency portfolio

Once we know ω_t, we directly know the portfolio:

$$F_{t+1} = \omega_t^\top R^e_{t+1},$$

which (by our earlier result) is the **max-Sharpe** portfolio given information at time t.

So we can **trade** it and evaluate its Sharpe, drawdowns, etc.



3) Decompose returns into systematic vs. idiosyncratic parts

With $\beta_{t,i}$ and F_{t+1}, for each stock:

$$R^e_{t+1,i} = \underbrace{\beta_{t,i}F_{t+1}}_{\text{predictable, systematic}} + \underbrace{\varepsilon_{t+1,i}}_{\text{unpredictable, idiosyncratic}}.$$

This helps with:

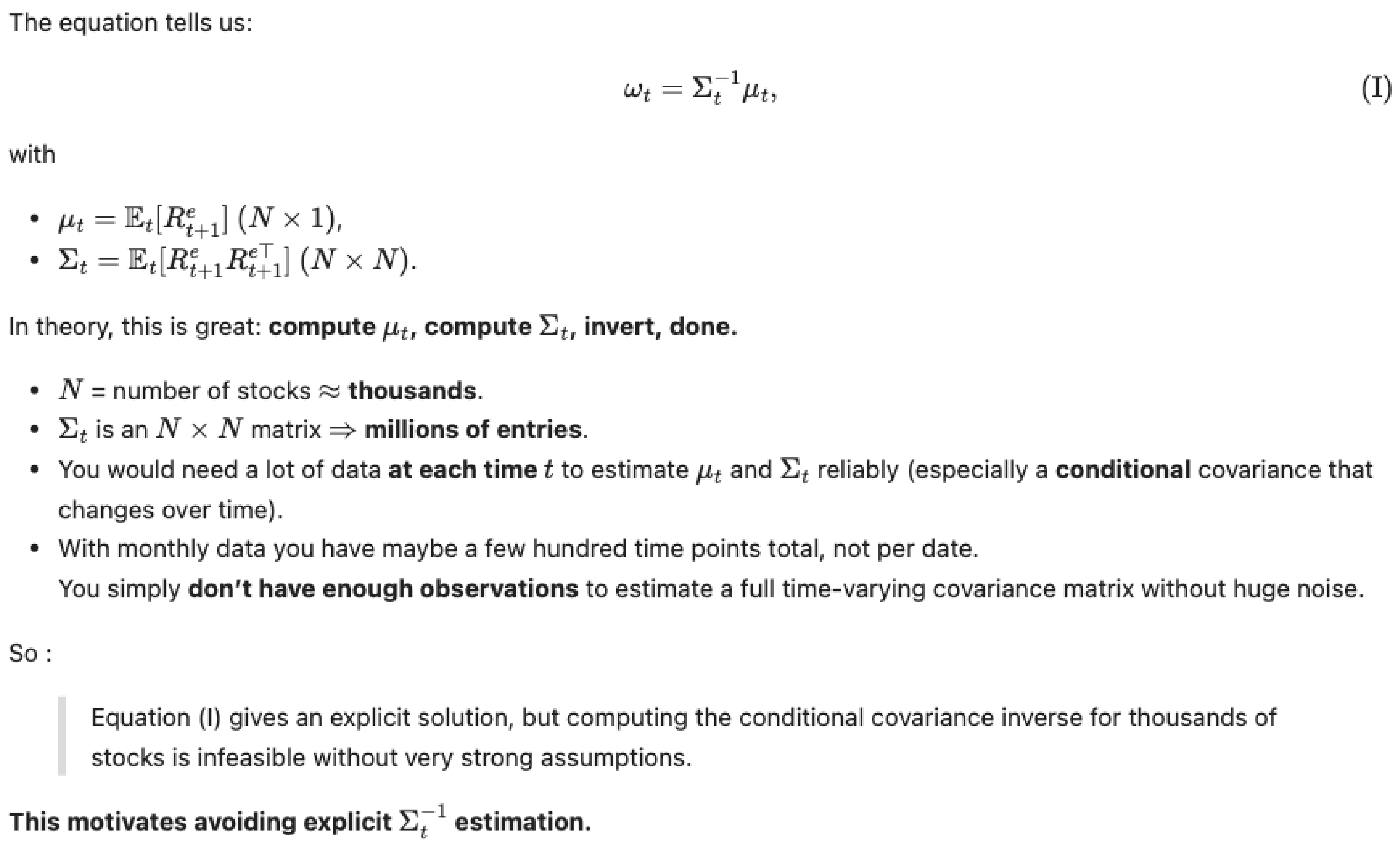
- **Explanation:** "this part is compensated risk"
- **Risk management:** "this part is diversifiable"

So the "three problems" are:

- **cross-section pricing / explanation** (via $\beta_{t,i}$),
- **optimal portfolio construction** (via ω_t),
- **risk decomposition** (via $\varepsilon_{t+1,i}$),

all solved once we can map $(I_t, I_{t,i}) \mapsto (\omega_t, \beta_{t,i})$.

Why the closed-form solution is infeasible in high dimensions



The equation tells us:

$$\omega_t = \Sigma_t^{-1} \mu_t, \tag{I}$$

with

- $\mu_t = \mathbb{E}_t[R_{t+1}^e]$ $(N \times 1)$,
- $\Sigma_t = \mathbb{E}_t[R_{t+1}^e R_{t+1}^{e\top}]$ $(N \times N)$.

In theory, this is great: **compute μ_t, compute Σ_t, invert, done.**

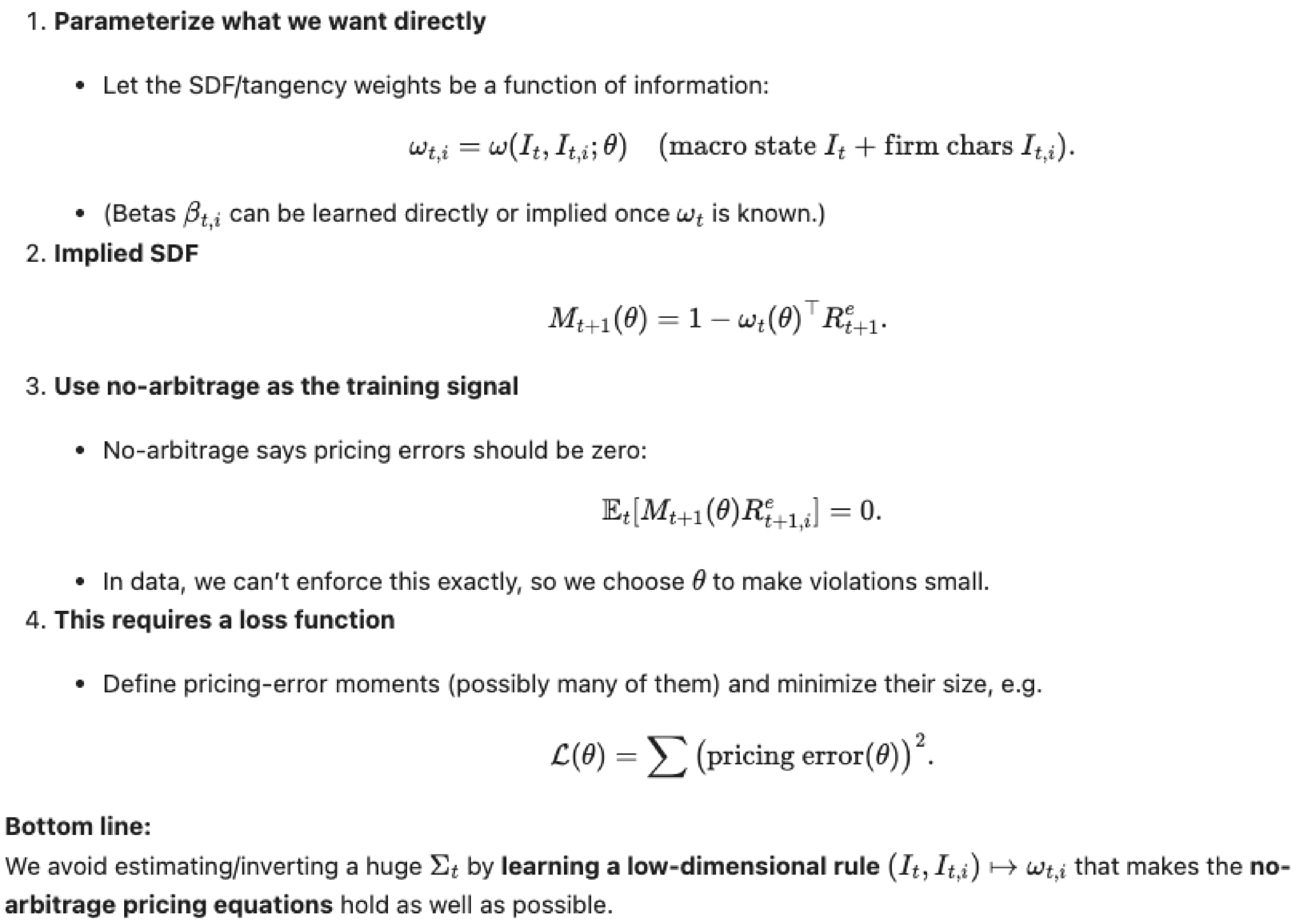
- N = number of stocks $\approx$ **thousands**.
- Σ_t is an $N \times N$ matrix $\Rightarrow$ **millions of entries.**
- You would need a lot of data **at each time** t to estimate μ_t and Σ_t reliably (especially a **conditional** covariance that changes over time).
- With monthly data you have maybe a few hundred time points total, not per date.
  You simply **don't have enough observations** to estimate a full time-varying covariance matrix without huge noise.

So :

> Equation (I) gives an explicit solution, but computing the conditional covariance inverse for thousands of stocks is infeasible without very strong assumptions.

This motivates avoiding explicit Σ_t^{-1} estimation.

New idea: guess a functional form for the weights, then fit it to satisfy no-arbitrage



1. **Parameterize what we want directly**
   - Let the SDF/tangency weights be a function of information:

$$\omega_{t,i} = \omega(I_t, I_{t,i}; \theta) \quad (\text{macro state } I_t + \text{firm chars } I_{t,i}).$$

   - (Betas $\beta_{t,i}$ can be learned directly or implied once ω_t is known.)

2. **Implied SDF**

$$M_{t+1}(\theta) = 1 - \omega_t(\theta)^\top R^e_{t+1}.$$

3. **Use no-arbitrage as the training signal**
   - No-arbitrage says pricing errors should be zero:

$$\mathbb{E}_t[M_{t+1}(\theta) R^e_{t+1,i}] = 0.$$

   - In data, we can't enforce this exactly, so we choose θ to make violations small.

4. **This requires a loss function**
   - Define pricing-error moments (possibly many of them) and minimize their size, e.g.

$$\mathcal{L}(\theta) = \sum \left(\text{pricing error}(\theta)\right)^2.$$

Bottom line:
We avoid estimating/inverting a huge Σ_t by **learning a low-dimensional rule** $(I_t, I_{t,i}) \mapsto \omega_{t,i}$ that makes the **no-arbitrage pricing equations** hold as well as possible.

From Conditioning to Unconditioning (the exact identity the paper uses)

The paper starts from a **conditional** no-arbitrage restriction:

$$\mathbb{E}[X \mid \mathcal{I}_t] = 0,$$

where in our application

$$X := M_{t+1} R^e_{t+1,i} \quad \text{and} \quad \mathcal{I}_t = (I_t, I_{t,i}) \text{ (macro + firm info at time } t\text{)}.$$

Now take **any function** $g(\mathcal{I}_t)$ that is measurable at time t (i.e., it depends only on information available at time t).

Key identity (Law of Iterated Expectations with instruments)

$$\mathbb{E}[g(\mathcal{I}_t)\, X] = \mathbb{E}\Big[g(\mathcal{I}_t)\, \mathbb{E}[X \mid \mathcal{I}_t]\Big].$$

So if $\mathbb{E}[X \mid \mathcal{I}_t] = 0$, then

$$\mathbb{E}[g(\mathcal{I}_t)\, X] = \mathbb{E}[g(\mathcal{I}_t) \cdot 0] = 0.$$

Applied to our problem

Start from the conditional pricing condition:

$$\mathbb{E}[M_{t+1} R^e_{t+1,i} \mid I_t, I_{t,i}] = 0.$$

Multiply by any instrument $g(I_t, I_{t,i})$ and take unconditional expectation:

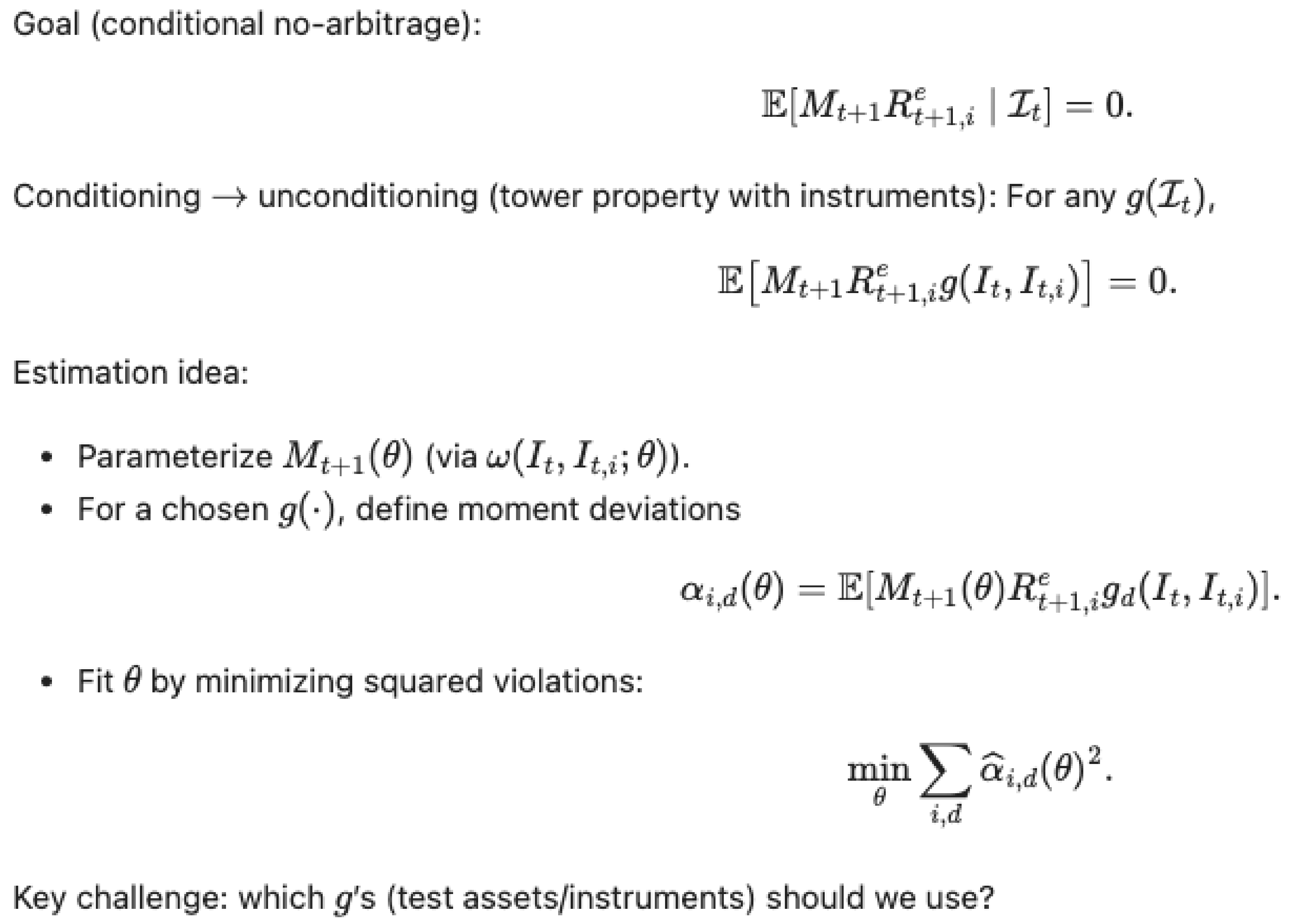
$$\boxed{\mathbb{E}[M_{t+1} R^e_{t+1,i}\, g(I_t, I_{t,i})] = 0 \quad \text{for any } g.}$$

Interpretation:

The conditional no-arbitrage condition implies **infinitely many unconditional moment conditions**, one for each choice of g.

These are exactly the moments the paper uses (and selects adversarially).

From conditional no-arbitrage to a trainable loss (GMM intuition)



Goal (conditional no-arbitrage):

$$\mathbb{E}[M_{t+1}R^e_{t+1,i} \mid \mathcal{I}_t] = 0.$$

Conditioning $\rightarrow$ unconditioning (tower property with instruments): For any $g(\mathcal{I}_t)$,

$$\mathbb{E}\big[M_{t+1}R^e_{t+1,i}g(I_t, I_{t,i})\big] = 0.$$

Estimation idea:

- Parameterize $M_{t+1}(\theta)$ (via $\omega(I_t, I_{t,i}; \theta)$).
- For a chosen $g(\cdot)$, define moment deviations

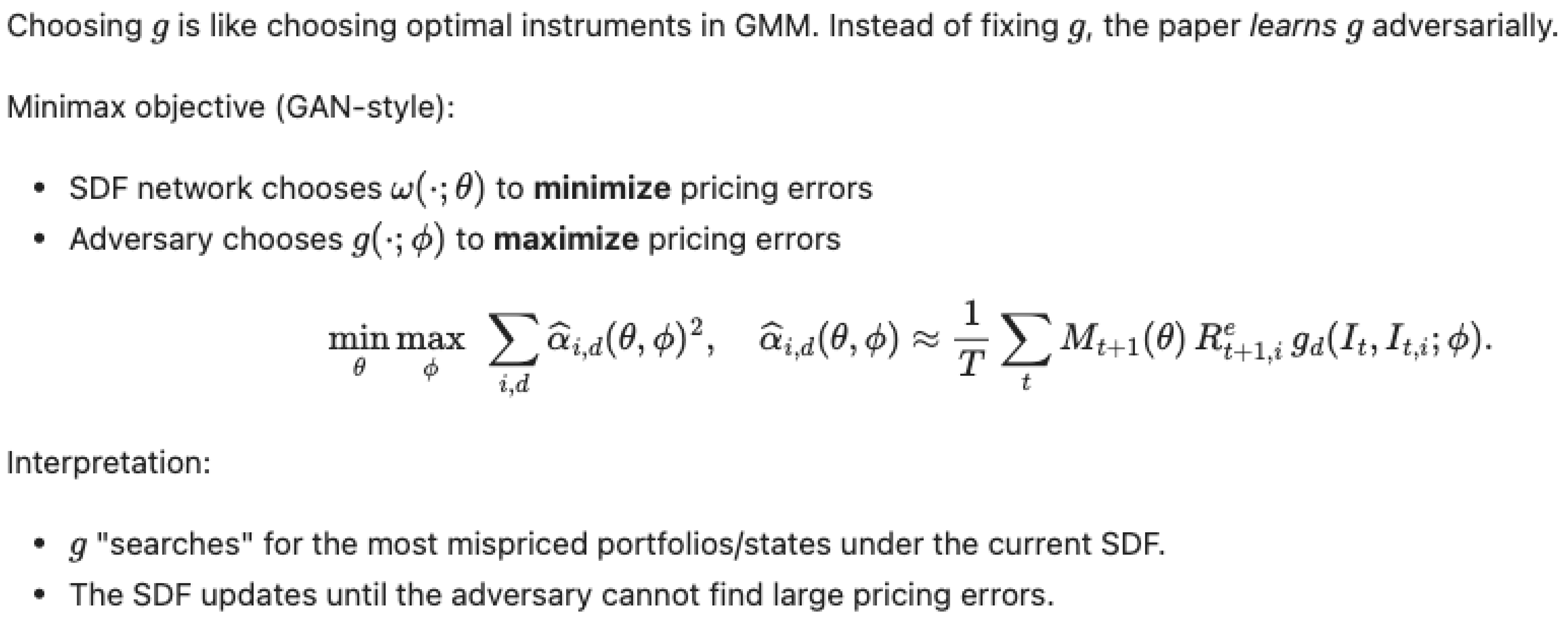
$$\alpha_{i,d}(\theta) = \mathbb{E}[M_{t+1}(\theta)R^e_{t+1,i}g_d(I_t, I_{t,i})].$$

- Fit θ by minimizing squared violations:

$$\min_{\theta} \sum_{i,d} \widehat{\alpha}_{i,d}(\theta)^2.$$

Key challenge: which g's (test assets/instruments) should we use?

From Choose g to a GAN-style minimax game



Choosing g is like choosing optimal instruments in GMM. Instead of fixing g, the paper *learns* g adversarially.

Minimax objective (GAN-style):

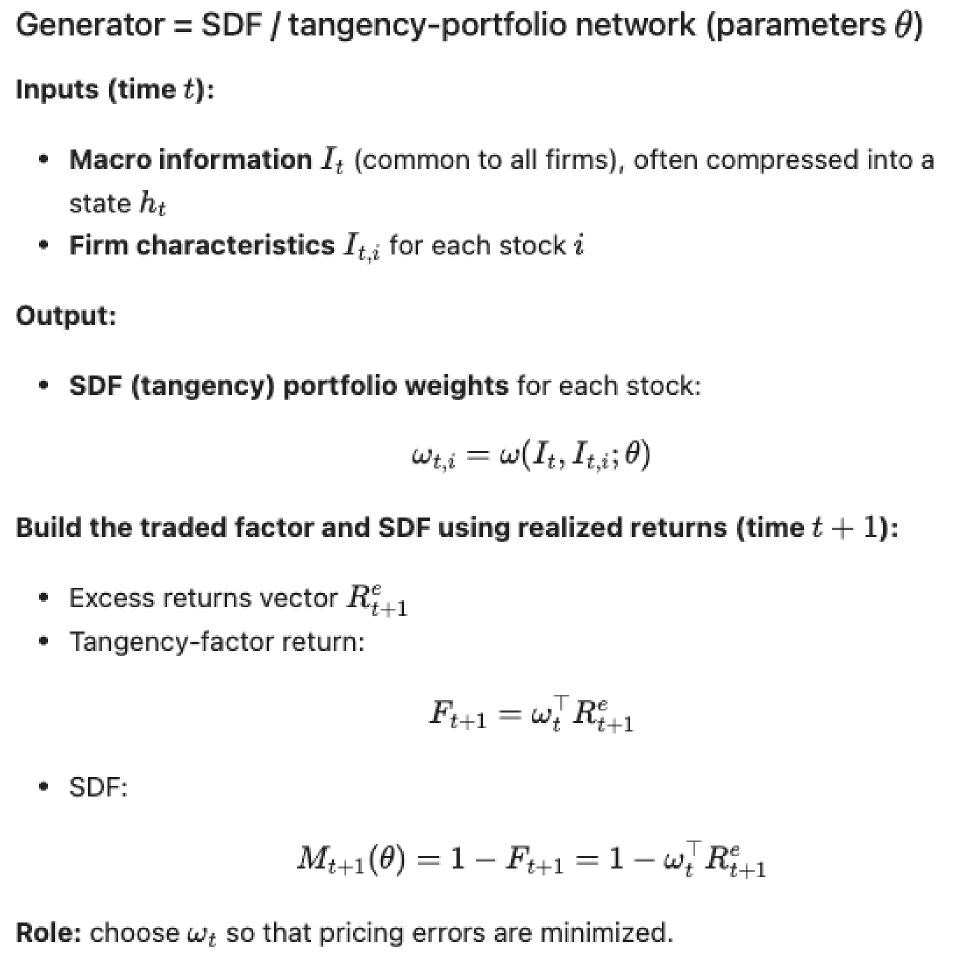
- SDF network chooses $\omega(\cdot;\theta)$ to **minimize** pricing errors
- Adversary chooses $g(\cdot;\phi)$ to **maximize** pricing errors

$$\min_{\theta}\max_{\phi}\ \sum_{i,d}\widehat{\alpha}_{i,d}(\theta,\phi)^2,\quad \widehat{\alpha}_{i,d}(\theta,\phi)\approx\frac{1}{T}\sum_{t}M_{t+1}(\theta)\,R^{e}_{t+1,i}\,g_d(I_t,I_{t,i};\phi).$$

Interpretation:

- g "searches" for the most mispriced portfolios/states under the current SDF.
- The SDF updates until the adversary cannot find large pricing errors.

GAN-style architecture: Generator (SDF) vs. Discriminator (test assets)



Generator = SDF / tangency-portfolio network (parameters θ)

Inputs (time t):

- **Macro information** I_t (common to all firms), often compressed into a state h_t
- **Firm characteristics** $I_{t,i}$ for each stock i

Output:

- **SDF (tangency) portfolio weights** for each stock:

$$\omega_{t,i} = \omega(I_t, I_{t,i}; \theta)$$

Build the traded factor and SDF using realized returns (time $t+1$):

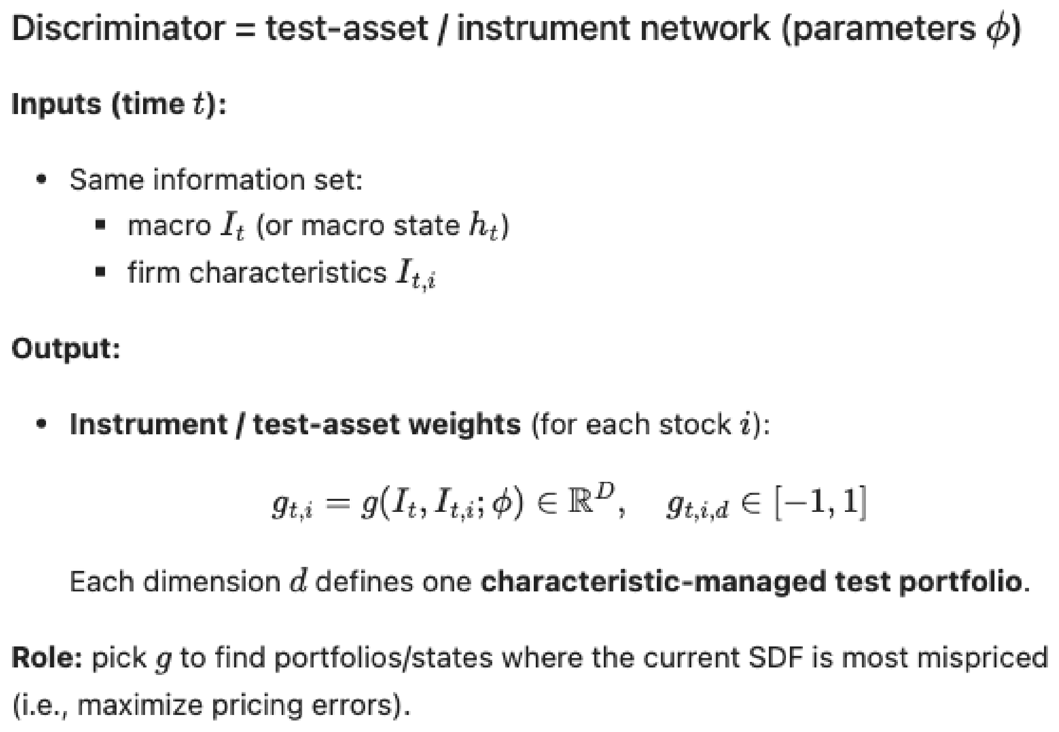
- Excess returns vector R^e_{t+1}
- Tangency-factor return:

$$F_{t+1} = \omega_t^\top R^e_{t+1}$$

- SDF:

$$M_{t+1}(\theta) = 1 - F_{t+1} = 1 - \omega_t^\top R^e_{t+1}$$

Role: choose ω_t so that pricing errors are minimized.



Discriminator = test-asset / instrument network (parameters ϕ)

Inputs (time t):

- Same information set:
  - macro I_t (or macro state h_t)
  - firm characteristics $I_{t,i}$

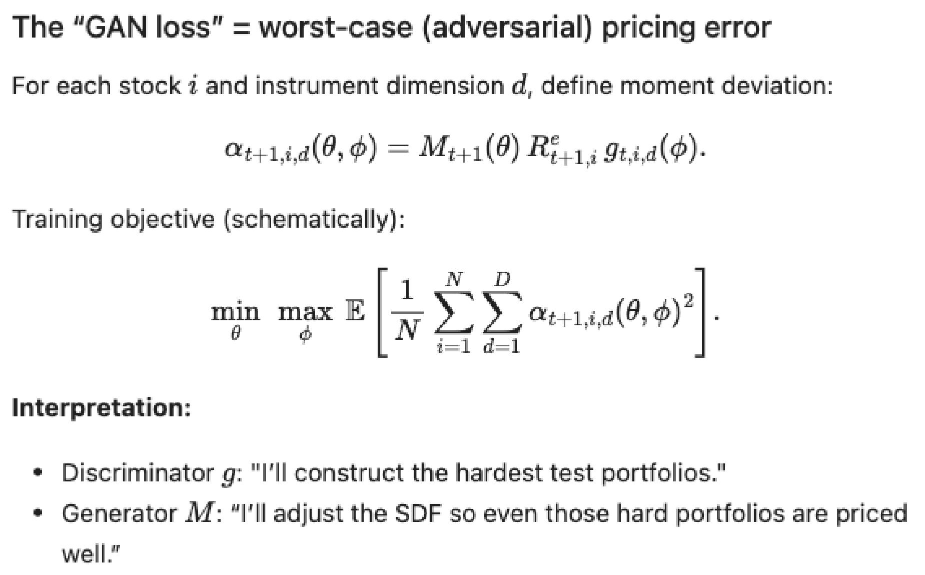
Output:

- **Instrument / test-asset weights** (for each stock i):

$$g_{t,i} = g(I_t, I_{t,i}; \phi) \in \mathbb{R}^D, \quad g_{t,i,d} \in [-1, 1]$$

Each dimension d defines one **characteristic-managed test portfolio**.

Role: pick g to find portfolios/states where the current SDF is most mispriced (i.e., maximize pricing errors).



The "GAN loss" = worst-case (adversarial) pricing error

For each stock i and instrument dimension d, define moment deviation:

$$\alpha_{t+1,i,d}(\theta, \phi) = M_{t+1}(\theta)\, R^e_{t+1,i}\, g_{t,i,d}(\phi).$$

Training objective (schematically):

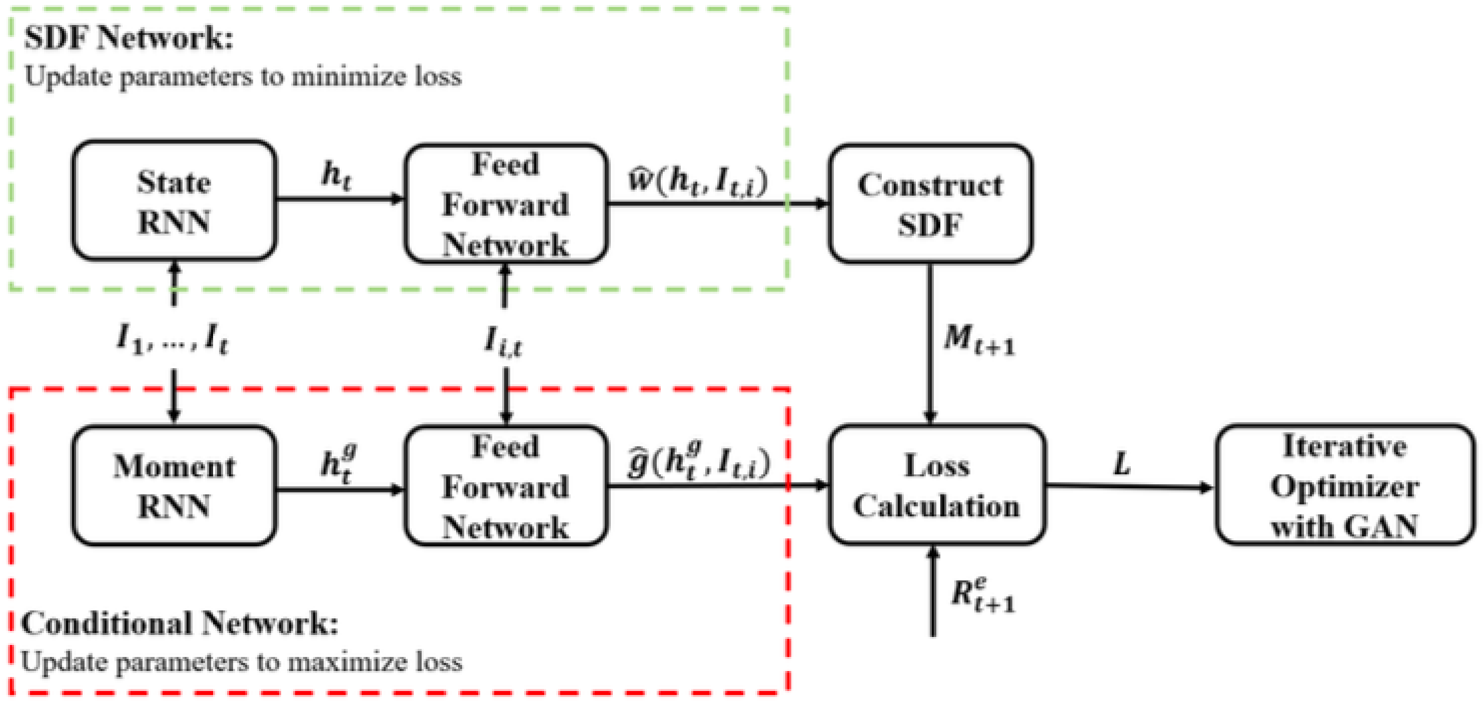
$$\min_\theta \max_\phi \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\sum_{d=1}^{D} \alpha_{t+1,i,d}(\theta, \phi)^2\right].$$

Interpretation:

- Discriminator g: "I'll construct the hardest test portfolios."
- Generator M: "I'll adjust the SDF so even those hard portfolios are priced well."

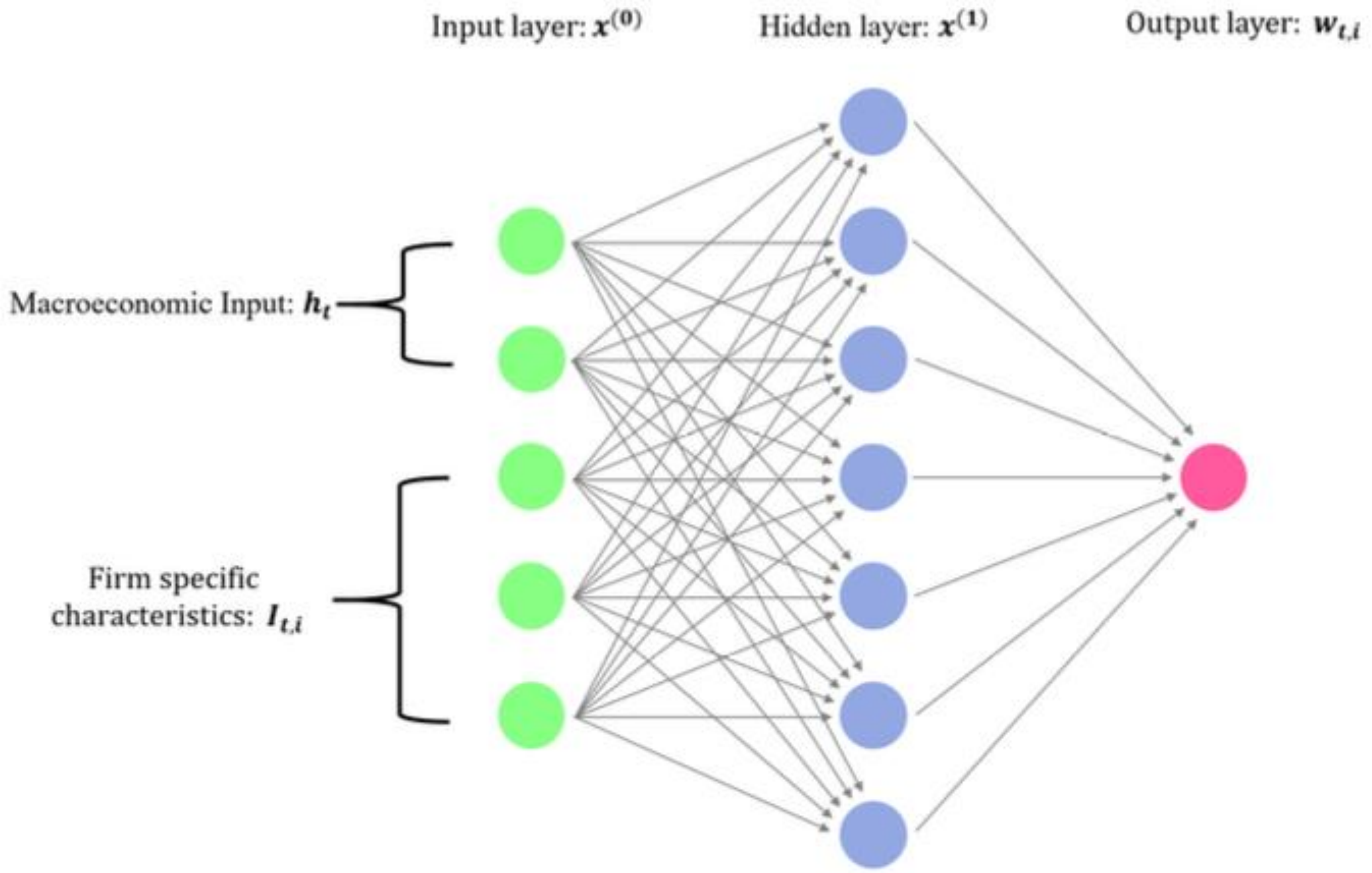
Architecture

Figure 1: GAN Model Architecture



This figures shows the model architecture of GAN (Generative Adversarial Network) with RNN (Recurrent Neural Network) with LSTM cells. The SDF network has two parts: (1) A LSTM estimates a small number of macroeconomic states. (2) These states together with the firm-characteristics are used in a FFN to construct a candidate SDF for a given set of test assets. The conditioning network also has two networks: (1) It creates its own set of macroeconomic states, (2) which it combines with the firm-characteristics in a FFN to find mispriced test assets for a given SDF M. These two networks compete until convergence, that is neither the SDF nor the test assets can be improved.

Figure 2: Illustration of Feedforward Network with Single Hidden Layer


Input layer: $\boldsymbol{x}^{(0)}$
Hidden layer: $\boldsymbol{x}^{(1)}$
Output layer: $\boldsymbol{w}_{t,i}$
Macroeconomic Input: $\boldsymbol{h}_t$
Firm specific characteristics: $\boldsymbol{I}_{t,i}$

Empirical Results for U.S. Equities

Returns & Sample Split

- **Universe:** all CRSP U.S. equities, **monthly** returns
- **Sample: Jan 1967 – Dec 2016** (50 years)
  - **Train:** 1967–1986 (20y)
  - **Validation:** 1987–1991 (5y)
  - **Out-of-sample test:** 1992–2016 (25y)
- **Risk-free rate:** 1-month T-bill (Kenneth French Data Library) $\rightarrow$ compute **excess returns**

Firm Characteristics (Cross-Section)

- **46 firm-specific characteristics** (from K. French library and/or Freyberger–Neuhierl–Weber, 2020)
- Built from **CRSP/Compustat** accounting variables + **past returns**
- **Updating convention:**
  - annual vars updated end of **June** (Fama–French convention)
  - monthly vars updated end of each **month** (used for next month)
- CRSP has ~**31,000** stocks, but requiring all characteristics each month leaves ~**10,000** stocks

Standardization & "Characteristic Factors"

- Each month: rank each characteristic **cross-sectionally** and convert to **quantiles**
- Linear projection form:

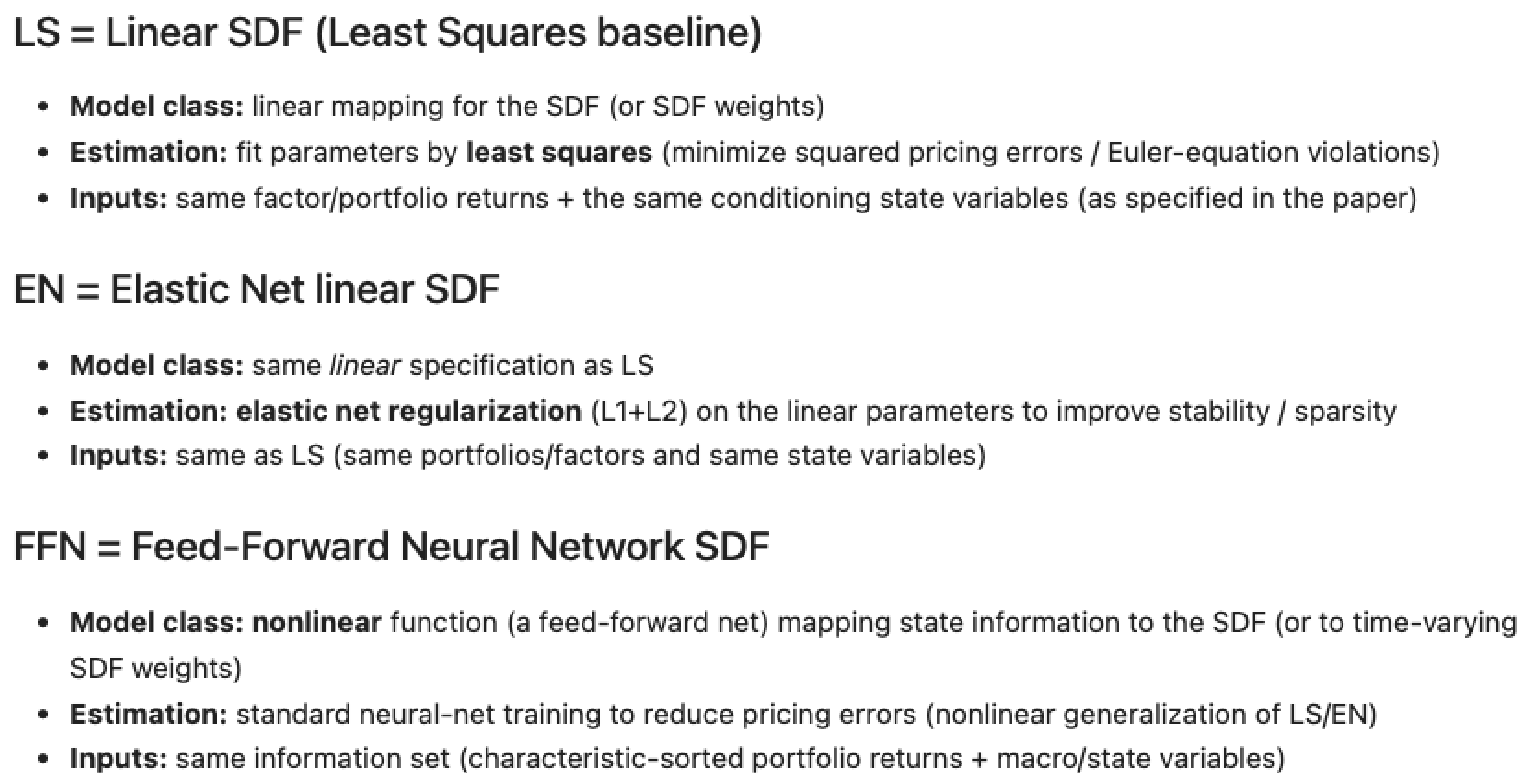
$$\hat{F}_{t+1} = \frac{1}{N}\sum_{i=1}^{N} I_{t,i} R_{t+1,i}$$

  $\rightarrow$ yields **long–short** characteristic factors (more weight above-median, less below-median)
- Increased flexibility: treat **positive and negative legs separately**
  - rank-weighted average of stocks **above** median + separately **below** median
  - **⇒ two "factors" per characteristic**
  - factors are **zero-cost** portfolios (built on excess returns)

Macroeconomic State Variables (Time Series)

- **178** macro predictors from 3 sources:
  - **124** from **FRED-MD** (McCracken & Ng, 2016)
  - **46** time series: cross-sectional **median** of each firm characteristic
  - **8** predictors from **Welch & Goyal (2007)** (not already in FRED-MD)
- Apply standard stationarity transformations:
  - FRED-MD transformations (McCracken & Ng)
  - analogous transformations for the 46 medians and the 8 Welch–Goyal series

Methods for comparison



LS = Linear SDF (Least Squares baseline)

- **Model class:** linear mapping for the SDF (or SDF weights)
- **Estimation:** fit parameters by **least squares** (minimize squared pricing errors / Euler-equation violations)
- **Inputs:** same factor/portfolio returns + the same conditioning state variables (as specified in the paper)

EN = Elastic Net linear SDF

- **Model class:** same *linear* specification as LS
- **Estimation: elastic net regularization** (L1+L2) on the linear parameters to improve stability / sparsity
- **Inputs:** same as LS (same portfolios/factors and same state variables)

FFN = Feed-Forward Neural Network SDF

- **Model class: nonlinear** function (a feed-forward net) mapping state information to the SDF (or to time-varying SDF weights)
- **Estimation:** standard neural-net training to reduce pricing errors (nonlinear generalization of LS/EN)
- **Inputs:** same information set (characteristic-sorted portfolio returns + macro/state variables)

Table I: Performance of Different SDF Models

	SR			EV			XS-R^2		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
LS	1.80	0.58	0.42	0.09	0.03	0.03	0.15	0.00	0.14
EN	1.37	1.15	0.50	0.12	0.05	0.04	0.17	0.02	0.19
FFN	0.45	0.42	0.44	0.11	0.04	0.04	0.14	-0.00	0.15
GAN	2.68	1.43	0.75	0.20	0.09	0.08	0.12	0.01	0.23

This table shows the monthly Sharpe ratio (SR) of the SDF, explained time series variation (EV) and cross-sectional mean R^2 for the GAN, FFN, EN and LS model.

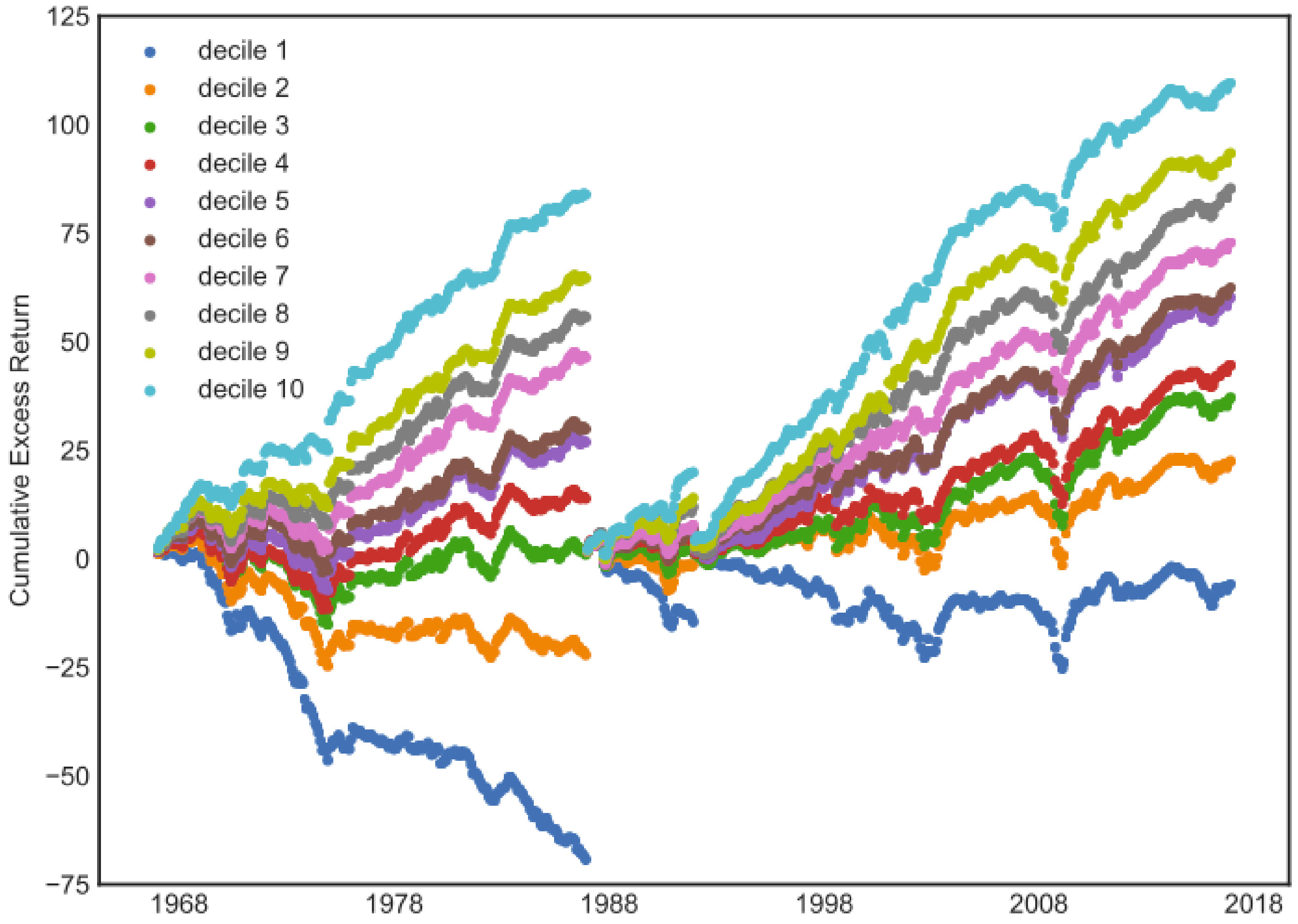
Table IV: Different SDF Models Evaluated on Large Market Cap Stocks

	SR			EV			Cross-Sectional R^2		
Model	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Evaluated for size $\geq 0.001\%$ of total market cap									
LS	1.44	0.31	0.13	0.07	0.05	0.03	0.14	0.03	0.10
EN	0.93	0.56	0.15	0.11	0.09	0.06	0.17	0.05	0.14
FFN	0.42	0.20	0.30	0.11	0.10	0.05	0.19	0.08	0.18
GAN	2.32	1.09	0.41	0.23	0.22	0.14	0.20	0.13	0.26
Evaluated for $\geq 0.01\%$ of total market cap									
LS	0.32	-0.11	-0.06	0.05	0.07	0.04	0.13	0.05	0.09
EN	0.37	0.26	0.23	0.09	0.12	0.07	0.17	0.08	0.14
FFN	0.32	0.17	0.24	0.13	0.22	0.09	0.22	0.15	0.26
GAN	0.97	0.54	0.26	0.28	0.34	0.18	0.27	0.23	0.32
Estimated and evaluated for size $\geq 0.001\%$ of total market cap									
LS	1.91	0.40	0.19	0.08	0.06	0.04	0.18	0.05	0.12
EN	1.34	0.92	0.42	0.13	0.13	0.07	0.23	0.09	0.19
FFN	0.37	0.19	0.28	0.13	0.13	0.07	0.21	0.10	0.21
GAN	3.57	1.18	0.42	0.24	0.23	0.14	0.23	0.13	0.26

The table shows monthly Sharpe ratios (SR) of the SDF factors, explained time series variation (EV) and cross-sectional R^2 for the GAN, FFN, EN and LS models. In the first two subtables the model is estimated on all stocks but evaluated on stocks with market capitalization larger than 0.01% or 0.001% of the total market capitalization. In the last subtable the model is estimated and evaluated on stocks market with capitalization larger than 0.001%.

Predictive Performance

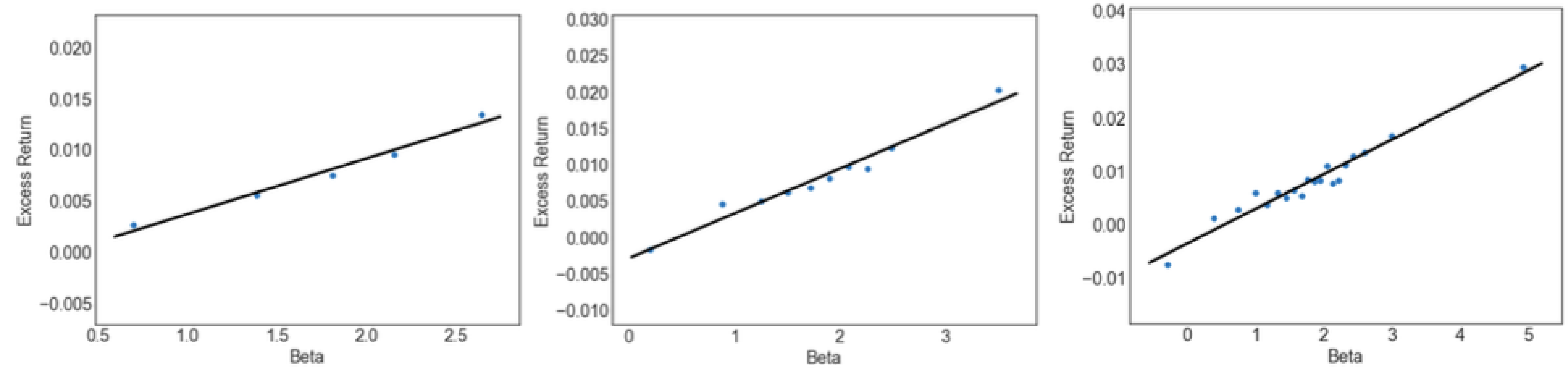
Figure 7: Cumulative Excess Return of Decile Sorted Portfolios with GAN



This figure shows the cumulative excess return of decile sorted portfolios based on the risk loadings β. The first portfolio is based on the smallest decile of risk loadings, while the last decile portfolio is constructed with the largest loading decile. Within each decile the stocks are equally weighted.

Predictive Performance

Figure 8: Expected Excess Returns of β-Sorted Portfolios as a Function of β



(a) 5 β-sorted quintiles **(b)** 10 β-sorted deciles **(c)** 20 β-sorted quantiles

This figure shows expected excess returns of β-sorted portfolios for GAN on the test sample. Stocks are sorted into 5, 10 or 20 quantiles every month. We plot them against the β of each portfolio that averages the individual stock $\beta_{t,i}$ over stocks and time within each portfolio. The linear line denotes a linear regression with intercept, which yields an R^2 of 0.98 (quintiles), 0.97 (deciles) and 0.95 (20 quantiles). The SDF itself has a β equal to one.

Replication repo

https://github.com/omroot/DeepLearningInAssetPricing_PaperRepl
ication

References

- Luyang Chen, Markus Pelger, Jason Zhu (2021), Deep Learning in Asset Pricing