BUSI 722

Session 7: Automated Trading & Risk Management

Kerry Back

Automated Trading

Automated Scheduled Trading

1. Download data at specified time of day (e.g., 30 minutes before market close)
2. Run model to generate trading signals
3. Run trading protocol to convert signals and current holdings to desired trades
4. Connect to broker API and submit trades
5. Receive execution confirmations

Real-Time Monitoring

- Stream live price data from broker
- Trigger alarms on extreme price movements
- Maybe trigger stop-loss orders

Risk Management

Risk Management

- Suppose you're a small cap fund manager.
- You will be evaluated relative to your peers or relative to Russell 2000.
- How do you measure risk?
- How do you manage risk?

Predicting Risk & Correlations

Why Predict Covariances?

- Portfolio optimization requires the covariance matrix Σ, not just expected returns μ.
- Minimum-variance portfolios need **only** Σ — no return forecasts at all.
- Covariances are **time-varying**: correlations spike in crises, volatility clusters.
- A good ML investing system should predict risks and correlations, not just returns.

Traditional Foundations

Univariate Volatility:

- **GARCH** (Bollerslev, 1986): $\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2$
- **Realized Volatility:** sum of squared intraday returns (from high-frequency data)
- **HAR-RV** (Corsi, 2009): predict daily RV from daily, weekly, and monthly lagged RV — simple but hard to beat

Multivariate:

- **DCC-GARCH** (Engle, 2002): scalable dynamic conditional correlations
- **Factor models** (Barra-style): $\Sigma = B\Lambda B' + D$
- **Shrinkage** (Ledoit-Wolf, 2004/2017): blend sample covariance with structured target to reduce estimation noise

ML for Volatility Prediction

Tree-Based Methods:

- Random forests and LightGBM applied to realized volatility forecasting
- Rich feature sets: lagged RV, trading volume, VIX, bid-ask spreads, macro variables
- Gains come from **nonlinear interactions** across features, not just lagged volatility

Deep Learning:

- **LSTM / BiLSTM:** capture long-range temporal dependencies in volatility series
- **GARCH-informed neural networks:** embed GARCH structure as initialization or constraint, then learn nonlinear extensions (ACM ICAIF, 2024)
- **Temporal Fusion Transformers:** multi-head attention over multiple time horizons with interpretable attention weights

ML for Covariance Matrices

Autoencoder Factor Models (Gu, Kelly, Xiu, 2021):

- Conditional autoencoders extract latent factors where both factors and loadings depend on asset characteristics
- Implied covariance: $\hat{\Sigma}_t = \hat{B}_t \hat{\Lambda}_t \hat{B}_t' + \hat{D}_t$

Graph-Based Methods (Zhang et al., *J. Financial Econometrics*, 2025):

- HAR model with neighborhood aggregation on graphs (sector, correlation-based)
- Exploits the network structure of asset relationships for covariance forecasts

Deep Learning Frameworks (Reis et al., *J. Forecasting*, 2025):

Practical Challenges

Positive Semi-Definiteness:

- ML-predicted $\hat{\Sigma}$ may not be PSD — enforce via Cholesky parameterization or projection

Dimensionality:

- 500 stocks $\Rightarrow$ 125,250 unique covariance entries
- Factor-model structure or graph-based methods are essential to reduce dimension

Non-Stationarity:

- Correlations are regime-dependent (spike in crises)

Portfolio Construction with Predicted Covariances

Minimum Variance:

- Requires only $\hat{\Sigma}$ — most sensitive to covariance quality
- ML-predicted covariances reduce realized portfolio variance vs. sample or DCC

Hierarchical Risk Parity (Lopez de Prado, 2016):

- Cluster assets using the correlation matrix, then allocate via recursive bisection
- No matrix inversion required — works even when $\hat{\Sigma}$ is singular
- More stable and diversified than Markowitz optimization out of sample

End-to-End Optimization:

Key References

- Corsi (2009), "A Simple Approximate Long-Memory Model of Realized Volatility," *J. Financial Econometrics*
- Engle (2002), "Dynamic Conditional Correlation," *JBES*
- Ledoit & Wolf (2017), "Nonlinear Shrinkage of the Covariance Matrix for Portfolio Selection," *RFS*
- Gu, Kelly, Xiu (2021), "Autoencoder Asset Pricing Models," *J. Econometrics*
- Bollerslev, Li, Patton, Quaedvlieg (2020), "Realized Semicovariances," *Econometrica*
- Zhang, Pu, Cucuringu, Dong (2025), "Graph-Based Methods for Forecasting Realized Covariances," *J. Financial Econometrics*
- Lopez de Prado (2016), "Building Diversified Portfolios that Outperform Out-of-Sample," *J. Portfolio Mgmt*

Direct Portfolio Construction from Signals

The Problem with Predict-Then-Optimize

Traditional approach:

1. Estimate expected returns $\hat{\mu}$ and covariance matrix $\hat{\Sigma}$
2. Feed $\hat{\mu}$ and $\hat{\Sigma}$ into a portfolio optimizer

Why this fails in practice:

- DeMiguel, Garlappi, Uppal (*RFS*, 2009): none of 14 optimized portfolios consistently beats equal weighting out of sample
- Need ∼3,000 months of data for mean-variance to beat $1/N$ with 25 assets
- Estimation error in $\hat{\mu}$ and $\hat{\Sigma}$ is **amplified** by the optimizer

Alternative: bypass moment estimation entirely and map signals directly to portfolio weights

Parametric Portfolio Policies

Brandt, Santa-Clara, Valkanov (*RFS*, 2009)

Portfolio weights are a **direct function of stock characteristics**:

$$w_{i,t} = \bar{w}_{i,t} + \frac{1}{N_t}\,\theta'\hat{x}_{i,t}$$

- $\bar{w}_{i,t}$ = benchmark weight (e.g., value-weighted market)
- $\hat{x}_{i,t}$ = cross-sectionally standardized characteristics (zero mean, unit std each month)
- θ = policy parameters estimated by maximizing expected CRRA utility

Key insight: estimate K parameters (one per characteristic) instead of N means + $N(N+1)/2$ covariances. With $K = 3$ (size, value, momentum), this is tractable for any

Principal Portfolios

Kelly, Malamud, Pedersen (*J. Finance*, 2023)

- Define a **prediction matrix** M where M_{ij} captures how signal j predicts return i — including **cross-asset predictability**.
- The optimal portfolio is the **eigenvector** of M (by analogy to PCA on the covariance matrix).
- Decompose M into symmetric and antisymmetric parts:
    - Symmetric $\Rightarrow$ **beta portfolios** (factor risk premia)
    - Antisymmetric $\Rightarrow$ **alpha portfolios** (market-neutral)
- Provides the **theoretical foundation** for why characteristic-based portfolios work.
- Brandt et al. is a special case (diagonal M, linear policy).

The Virtue of Complexity

Kelly, Malamud, Zhou (*J. Finance*, 2024)

- Proves that **more complex models generically outperform simpler ones** for return prediction and portfolio construction.
- Out-of-sample R^2 and optimal Sharpe ratio **increase with model parameterization**, even when parameters exceed observations.
- Holds in extremely data-scarce environments ($<$ 20 observations, tens of thousands of predictors).
- ML strategies learn to divest before recessions — successful in 14 of 15 NBER recessions out of sample.

Practical implication: use the largest model you can compute.

Random Fourier Features + Ridge Regression

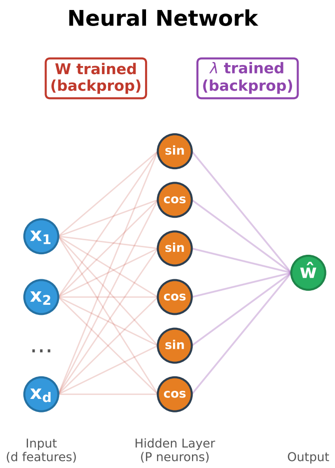
Didisheim, Ke, Kelly, Malamud (2024)

The idea: convert d raw characteristics X_t into P nonlinear features via random projections:

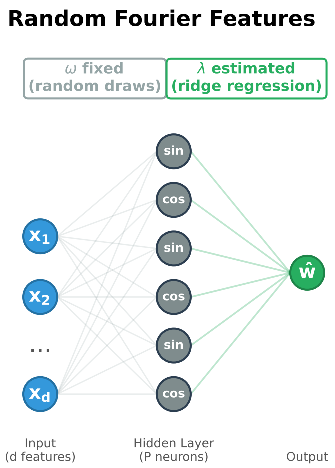
$$S_{i,t} = \left[\sin(\gamma X_t \omega_i),\ \cos(\gamma X_t \omega_i)\right]', \quad \omega_i \sim \text{i.i.d. } \mathcal{N}(0, I)$$

- Equivalent to a wide two-layer neural network with **fixed random weights** in the first layer
- Only the second layer (portfolio weights λ) is estimated — by **ridge regression**
- Can scale P from 2 to 1,000,000 from the same raw data

RFF ≈ Neural Network with Frozen First Layer



All weights optimized jointly



Only output weights estimated (closed-form solution)

Random Fourier Features: Results

Key findings (Didisheim et al., 2024):

- Out-of-sample SDF Sharpe ratio rises from $\sim$1 (low complexity) to $\sim$4 (high complexity) — far exceeding Fama-French 6-factor model (SR $\approx$ 1.1)
- Out-of-sample pricing errors fall by a factor of six as P grows
- Even using only the 5 Fama-French characteristics, adding nonlinear complexity **doubles the Sharpe ratio** and cuts pricing errors by half
- Robust across market cap groups (mega, large, small, micro)

Why it matters: achieves performance rivaling deep learning with **no backpropagation** — just random projections + ridge regression.

Deep Learning Extensions

Deep Parametric Portfolio Policies (Simon, Weibels, Zimmermann, 2023):

- Replace Brandt et al.'s linear weight function with a **feed-forward neural network**
- Captures nonlinear interactions among characteristics
- 75–276 bps/month improvement in certainty equivalent returns
- Risk aversion γ serves as an economically motivated regularization parameter

AI Asset Pricing Models (Kelly, Kuznetsov, Malamud, Xu, 2025):

- Embeds a **transformer** in the stochastic discount factor
- **Cross-asset attention** captures interactions across the full stock universe
- Large reductions in pricing errors vs. previous ML models
- Current frontier of the field

References: Direct Portfolio Construction

- DeMiguel, Garlappi, Uppal (2009), "Optimal Versus Naive Diversification," *RFS*
- Brandt, Santa-Clara, Valkanov (2009), "Parametric Portfolio Policies," *RFS*
- Kelly, Malamud, Pedersen (2023), "Principal Portfolios," *J. Finance*
- Kelly, Malamud, Zhou (2024), "The Virtue of Complexity in Return Prediction," *J. Finance*
- Didisheim, Ke, Kelly, Malamud (2024), "Complexity in Factor Pricing Models," Working Paper
- Simon, Weibels, Zimmermann (2023), "Deep Parametric Portfolio Policies," CFR Working Paper
- Kelly, Kuznetsov, Malamud, Xu (2025), "Artificial Intelligence Asset Pricing Models," NBER WP 33351
- Gu, Kelly, Xiu (2020), "Empirical Asset Pricing via Machine Learning," *RFS*

Using Apps and Scripts with AI

- Where possible, permanently fix the code used for tasks rather than asking AI to regenerate it each time.
- This ensures consistent behavior and reduces token usage.
- Save as a .py file and use in a Claude Code skill.