Technical Trading: Fools Gold?
Forecast Evaluation with Many Forecasts

The Econometrics of Predictability

This version: June 4, 2014

May 2014

UNIVERSITY OF
OXFORD

Overview

- Bootstrap
- Constructing Technical Trading Rules
- Multiple Hypothesis Testing
  - Reality Check
  - Hansen's Test of Superior Predictive Ability
  - Bonferroni and Bonferroni-Holm Bounds
  - StepM
  - Model Confidence Set
  - False Discovery Rate Control

UNIVERSITY OF OXFORD

Definition (The Bootstrap)

The bootstrap is a statistical procedure where data is resampled, and the resampled data is used to estimate quantities of interest.

- Bootstraps come in many forms
  - Structure
    - Parametric
    - Nonparametric
  - Dependence Type
    - IID
    - Wild
    - Block and other for dependent data
- All share common structure of using simulated random numbers in combination with original data to compute quantities of interest
- Applications
  - Confidence Intervals
  - Refinements
  - Bias estimation

Basic Problem

- Compute standard deviation for an estimator
- For example, in case of mean $\bar{x}$ for i.i.d. data, we know

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

  is usually a reasonable estimator of the standard deviation of the data
- The standard error of the mean is then

$$\mathrm{V}[\bar{x}] = \frac{s^2}{n}$$

  which can be used to form confidence intervals or conduct hypothesis tests (in conjunction with CLT)
- How could you estimate the standard error for the median of $x_1, \ldots x_n$?
- What about inference about a quantile, for example that 5^{th} percentile of $x_1, \ldots x_n$?
- Bootstrap is a computational method to construct standard error estimates of confidence interval for a wide range of estimators.

IID Bootstrap

- Assume n i.i.d. random (possibly vector valued) variables $\mathbf{x}_1, \ldots, \mathbf{x}_n$
- Estimator of of a parameter of interest $\hat{\theta}$
  - For example, the mean

Definition (Empirical Distribution Function)

The empirical distribution function assigns probability $1/n$ to each observation value. For a scalar random variable x_i, $i = 1, \ldots, n$, the EDF is defined

$$\hat{F}(X) = \frac{1}{n} \sum_{i=1}^{n} I_{[x_i < X]}.$$

- Also known as the empirical CDF
- CDF of X should have information about precision of $\hat{\theta}$, so ECDF might also

UNIVERSITY OF OXFORD

Algorithm (IID Bootstrap)

1. *Simulate a set of n i.i.d. uniform random integers* u_i, $i = 1, \ldots n$ *from the range* $1, \ldots, n$ *(with replacement)*
2. *Construct a bootstrap sample* $x_b^\star = \{x_{u_1}, x_{u_2}, \ldots x_{u_n}\}$
3. *Compute the mean*

$$\hat{\theta}_b^\star = \frac{1}{n} \sum_{i=1}^{n} x_{b,i}^\star$$

4. *Repeat steps 1–3* B *times*
5. *Estimate the standard of* $\hat{\theta}$ *using*

$$\frac{1}{B} \sum_{i=1}^{B} \left(\theta_b^\star - \hat{\theta}\right)^2$$

MATLAB Code for IID Bootstrap

UNIVERSITY OF OXFORD

```
n = 100; x = randn(n,1);
% Mean of x
mu = mean(x);
B = 1000;
% Initialize muStar
muStar = zeros(B,1);
% Loop over B bootstraps
for b=1:B
    % Uniform random numbers over 1...n
    u = ceil(n*rand(n,1));
    % x-star sample simulation
    xStar = x(u);
    % Mean of x-star
    muStar(b) = mean(xStar);
end
s2 = 1/(n-1)*sum((x-mu).^2);
stdErr = s2/n
bootstrapStdErr = mean((muStar-mu).^2)
```

How many bootstrap replications?

UNIVERSITY OF OXFORD

- B is used for the number of bootstrap replications
- Bootstrap theory assumes $B \to \infty$ quickly
- This ensures that the bootstrap distribution is identical to the case where all unique bootstraps were computed
  - There are a lot of unique bootstraps
  - n^n in the i.i.d. case
- Using finite B adds some extra variation since two bootstraps with the same data won't produce identical estimates
- Note: Often useful to set the state of your random number generator so that results are reproducible

```
% A non-negative integer
seed = 26031974
rng(seed)
```

- Should choose B large enough that the *Monte Carlo error* is negligible
- In practice little reason to use less than $1,000$ replications

Getting the most out of B bootstrap replications

UNIVERSITY OF OXFORD

- Balanced resampling
  - In standard i.i.d. bootstrap, some values will inevitibly appear more than others
  - Balanced resampling ensures that all values appear the same number of times
  - In practice simple to implement

Algorithm (IID Bootstrap with Balanced Resampling)

1. *Replicate the data so that there are B copies of each x_i. The data set should have Bn observations*
2. *Construct a random random permutation of the numbers $1, \ldots, Bn$ as $u_1, \ldots u_{Bn}$*
3. *Construct the bootstrap sample $x_b^\star = \left\{x_{u_{n(b-1)+1}}, x_{u_{n(b-1)+2}}, \ldots x_{u_{n(b-1)+n}}\right\}$*

- This algorithm samples *without replacement* from the replicated dataset of Bn observations
- Each data point will appear exactly B times in the B bootstrap samples

MATLAB Code for IID Balanced Bootstrap

UNIVERSITY OF OXFORD

```
n = 100; x = randn(n,1);
% Replicate the data
xRepl = repmat(x,B,1);
B = 1000;
% Random permutaiton of 1,...,B*n
u = randperm(n*B);
% Loop over B bootstraps
for b=1:B
    % Uniform random numbers over 1...n
    ind = n*(b-1)+(1:n);
    xb = xRepl(u(ind));
end
```

Getting the most out of B bootstrap replications

UNIVERSITY OF OXFORD

- Antithetic Random Variables
- If samples are *negatively* correlated, variance of statistics can be reduced
  - Basic idea is to order data so that if one sample has too many large values of x, then the next will have too many small
  - This can induce negative correlation while not corrupting bootstrap

Algorithm (IID Bootstrap with Antithetic Resampling)

1. *Order the data so that* $x_1 \leq x_2 \ldots \leq x_n$. *Treat these indices as the original data.*
2. *Simulate a set of* n *i.i.d. uniform random integers* u_i, $i = 1, \ldots n$ *from the range* $1, \ldots, n$ *(with replacement)*
3. *Construct the bootstrap sample* $x_b^\star = \{x_{u_1}, x_{u_2}, \ldots x_{u_n}\}$
4. *Construct* $\tilde{u}_i = n - u_i + 1$
5. *Construct the antithetic bootstrap sample* $x_{b+1}^\star = \{x_{\tilde{u}_1}, x_{\tilde{u}_2}, \ldots x_{\tilde{u}_n}\}$
6. *Repeat for* $b = 1, 3, \ldots, B - 1$

- Using antithetic random variables is a general principle applicable to virtually all simulation estimators

MATLAB Code for IID Bootstrap with Antithetic RV

```
n = 100; x = randn(n,1);
% Mean of x
mu = mean(x);
B = 1000;
% Initialize muStar
muStar = zeros(B,1);
% Sort x
x = sort(x);
% Loop over B bootstraps
for b=1:2:B
    % Uniform random numbers over 1...n
    u = ceil(n*rand(n,1)); xStar = x(u);
    % Mean of x-star
    muStar(b) = mean(xStar);
    % Uniform random numbers over 1...n
    u = n-u+1; xStar = x(u);
    % Mean of x-star
    muStar(b+1) = mean(xStar);
end
corr(muStar(1:2:B),muStar(2:2:B))
```

Bootstrap Estimation of Bias

- Many statistics have a *finite sample bias*
- This is equivalent to saying that $\hat{\theta} - \theta \approx c/n$ for some $c \neq 0$
  - Many estimators have $c = 0$, for example the sample mean
  - These estimators are unbiased
- Biased estimators usually arise when the estimator is a non-linear function of the data
- Bootstrap can be used to estimate the bias, and the estimate can be used to debias the original estimate
- Recall the definition of bias

Definition (Bias)

The bias of an estimator is

$$\mathrm{E}\left[\hat{\theta} - \theta\right]$$

Bootstrap Estimation of Bias

Algorithm

1. *Estimate the parameter of interest* $\hat{\theta}$
2. *Generate a bootstrap sample* x_b *and estimate the parameter on the bootstrap sample. Denote this estimate as* $\hat{\theta}_b^\star$
3. *Repeat 2 a total of* B *times*
4. *Estimate the bias as*

$$Bias = B^{-1} \sum_{i=1}^{B} \hat{\theta}_b^\star - \hat{\theta}$$

- Example of bootstrap bias adjustment will be given later once more results for time-series have been established

Bootstrap Estimation of Standard Error

UNIVERSITY OF OXFORD

Algorithm

1. *Estimate the parameter of interest* $\hat{\theta}$
2. *Generate a bootstrap sample* x_b *and estimate the parameter on the bootstrap sample. Denote this estimate as* $\hat{\theta}_b^\star$
3. *Repeat 2 a total of* B *times*
4. *Estimate the standard error as*

$$\textit{Std. Err} = \sqrt{B^{-1} \sum_{i=1}^{B} \left(\hat{\theta}_b^\star - \hat{\theta}\right)^2}$$

- Other estimators are also common

$$\text{Std. Err} = \sqrt{(B-1)^{-1} \sum_{i=1}^{B} \left(\hat{\theta}_b^\star - \overline{\hat{\theta}_b^\star}\right)^2}$$

- B should be sufficiently large that B or $B-1$ should not matter

Bootstrap Estimation of Confidence Intervals

UNIVERSITY OF OXFORD

- Bootstraps can also be used to construct confidence intervals
- Two methods:
  1. Estimate the standard error of the estimator and use a CLT
  2. Estimate the confidence interval directly using the bootstrap estimators $\left\{\hat{\theta}_b^\star\right\}$
- The first method is simple and have previously been explained
- The second is also very simple, and is known as the *percentile method*

Percentile Method

Algorithm (Percentile Method)

A confidence interval $\left[C_{\alpha_L}, C_{\alpha_H}\right]$ *with coverage* $\alpha_H - \alpha_L$ *can be constructed:*

1. *Construct a bootstrap sample* x_b
2. *Compute the bootstrap estimate* $\hat{\theta}_b^\star$
3. *Repeat steps 1–2*
4. *The confidence interval is constructed using the empirical* α_L *quantile and the empirical* α_H *quantile of* $\left\{\hat{\theta}_b^\star\right\}$

- If the bootstrap estimates are ordered from smallest to largest, and $B\alpha_L$ and $B\alpha_H$ are integers, then the confidence interval is

$$\left[\hat{\theta}^\star_{B\alpha_L}, \hat{\theta}^\star_{B\alpha_H}\right]$$

- This method may not work well in all situations
  - n small
  - Highly asymmetric distribution

MATLAB Code for Percentile Method

UNIVERSITY OF OXFORD

```
n = 100; x = randn(n,1);
% Mean of x
mu = mean(x);
B = 1000;
% Initialize muStar
muStar = zeros(B,1);
% Loop over B bootstraps
for b=1:B
    % Uniform random numbers over 1...n
    u = ceil(n*rand(n,1));
    % x-star sample simulation
    xStar = x(u);
    % Mean of x-star
    muStar(b) = mean(xStar);
end
alphaL = .05;alphaH=.95;
muStar = sort(muStar);
CI = [muStar(alphaL*B) muStar(round(alphaH*B))]
CI - mu
```

Bootstrap and Regression

- Bootstraps can be used in more complex scenarios
- One simple extension is to regressions
- Using a model, rather than estimating a simple statistic, allows for a richer set of bootstrap options
  - Parametric
  - Non-parametric
- Basic idea, however, remains the same:
  - Simulate random data from the same DGP
  - Now requires data for both the regressor y and the regressand $\mathbf{x}$

Parametric vs. Non-parametric Bootstrap

UNIVERSITY OF OXFORD

- Parametric bootstraps are based on a model
- They exploit the structure of the model to re-sample residuals rather than the actual data
- Suppose

$$y_i = \mathbf{x}_i \boldsymbol{\beta} + \epsilon_i$$

  where ϵ_i is homoskedastic
- The parametric bootstrap would estimate the model and the residuals as

$$\hat{\epsilon}_i = y_i - \mathbf{x}_i \hat{\boldsymbol{\beta}}$$

- The bootstrap would then construct the re-sampled "data" by sampling $\hat{\epsilon}_i$ separately from $\mathbf{x}_i$
  - In other words, use two separate sets of i.i.d. uniform indices
- Construct $y^{\star}_{b,i} = \mathbf{x}_{u_{1i}} \hat{\boldsymbol{\beta}} + \hat{\epsilon}_{u_{2i}}$
- Compute statistics using these values

Useful function: bsxfun

- Many examples use bsxfun

```
BSXFUN  Binary Singleton Expansion Function

C = BSXFUN(FUNC,A,B) applies the element-by-element binary
operation specified by the function handle FUNC to arrays A and
B, with singleton expansion enabled. FUNC must be able to accept
as input either two column vectors of the same size, or one
column vector and one scalar, and return as output a column
vector of the same size as the input(s).  FUNC can either be a
function handle for an arbitrary function satisfying the above
conditions or one of the following built-in:
```

- Allows k by n matrix to be added/subtracted from k by 1 vector or 1 by n vector

```
x = randn(1000,10);
mu = mean(x);
err = bsxfun(@minus,x,mu);
```

MATLAB Code for Parametric Bootstrap of Regression

```
n = 100; x = randn(n,2); e = randn(n,1); y = x*ones(2,1) + e;
% Bhat
Bhat = x\y; ehat = y - x*Bhat;
B = 1000;
% Initialize BStar
BStar = zeros(B,2);
% Loop over B bootstraps
for b=1:B
    % Uniform random numbers over 1...n
    uX = ceil(n*rand(n,1)); uE = ceil(n*rand(n,1));
    % x-star sample simulation
    xStar = x(uX,:); eStar = e(uE);
    yStar = xStar*Bhat + eStar;
    % Mean of x-star
    BStar(b,:) = (xStar\yStar)';
end
Berr=bsxfun(@minus,BStar,Bhat');
bootstrapVCV = Berr'*Berr/B
trueVCV = eye(2)/100
OLSVCV = (e'*e)/n * inv(x'*x)
```

Non-parametric Bootstrap

- Non-parametric bootstrap is simpler
- It does not use the structure of the model to construct artificial data
- The vector $[y_i, \mathbf{x}_i]$ is instead directly re-sampled
- The parameters are constructed from the pairs

Algorithm (Non-parametric Bootstrap for i.i.d. Regression Data)

1. *Simulate a set of n i.i.d. uniform random integers* u_i, $i = 1, \ldots n$ *from the range* $1, \ldots, n$ *(with replacement)*
2. *Construct the bootstrap sample* $\mathbf{z}_b = \{y_{u_i}, \mathbf{x}_{u_i}\}$
3. *Estimate the bootstrap* $\boldsymbol{\beta}$ *by fitting the model*

$$y_{u_i} = \mathbf{x}_{u_i} \hat{\boldsymbol{\beta}}^{\star}_b + \epsilon^{\star}_{b,i}$$

MATLAB Code for Nonparametric Bootstrap of Regression

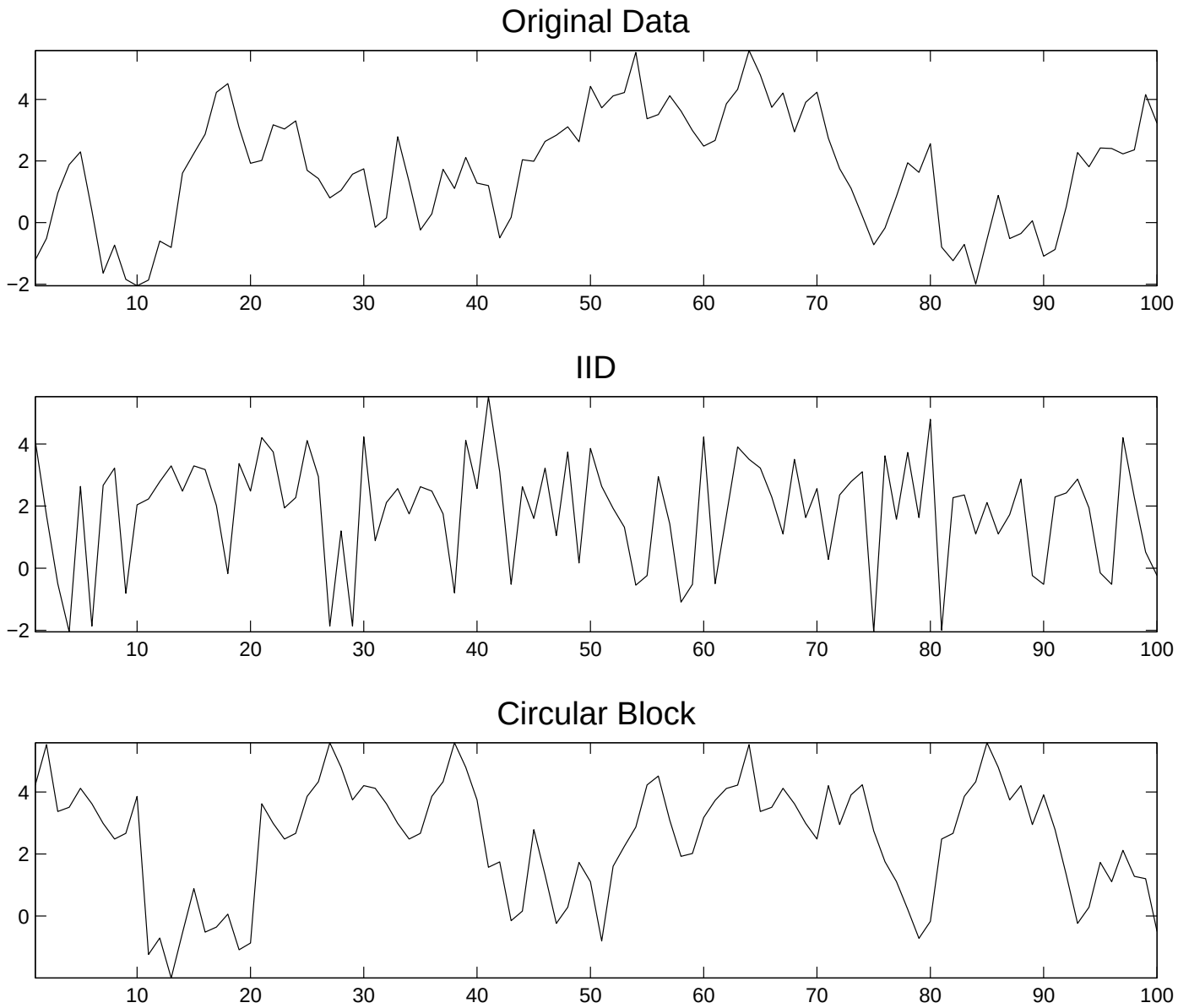
```
n = 100; x = randn(n,2); e = randn(n,1); y = x*ones(2,1) + e;
% Bhat
Bhat = x\y; ehat = y - x*Bhat;
B = 1000;
% Initialize BStar
BStar = zeros(B,2);
% Loop over B bootstraps
for b=1:B
    % Uniform random numbers over 1...n
    u = ceil(n*rand(n,1));
    % x-star sample simulation
    yStar = y(u);
    xStar = x(u,:);
    % Mean of x-star
    BStar(b,:) = (xStar\yStar)';
end
Berr=bsxfun(@minus,BStar,Bhat');
bootstrapVCV = Berr'*Berr/B
trueVCV = eye(2)/100
OLSVCV = (e'*e)/n * inv(x'*x)
```

Bootstrapping Time-series Data

UNIVERSITY OF OXFORD

- i.i.d. bootstrap is only appropriate for i.i.d. data
  - Note: Usually OK for data that is not serially correlated
- Two strategies for bootstrapping time-series data
  - Parametric & i.i.d. bootstrap: If the model postulates that the residuals are i.i.d. or at least white noise, then a residual-based i.i.d. bootstrap may be appropriate
    - Examples: AR models, GARCH models using appropriately standardized residuals
  - Nonparametric *block* bootstrap: Weak assumptions, basically that blocks can be sampled so that they (blocks) are approximately i.i.d.
    - Similar to the notion of ergodicity which is related to asymptotic independence
    - Important: Like Newey-West covariance estimator, *block length* must grow with sample size
    - Fundamentally same reason

The problem with the IID Bootstrap



MATLAB Code for all time-series applications

UNIVERSITY OF OXFORD

```
% Number of time periods
T = 100;
% Random errors
e = randn(T,1);
y = zeros(T,1);
% Y is an AR(1), phi1 = 0.5
y(1) = e(1)*sqrt(1/(1-.5^2));
for t=2:T
    y(t)=0.5*y(t-1)+e(t);
end
% 10,000 replications
B = 10000;
% Initial place for mu-star
muStar = zeros(B,1);
```

Moving Block Bootstrap

- Samples blocks of m consecutive observations
- Uses blocks which start at indices $1, \dots T - m + 1$

Algorithm (Moving Block Bootstrap)

1. *Initialize* $i = 1$
2. *Draw a uniform integer* v_i *on* $1, \dots, T - m + 1$
3. *Assign* $u_{(i-1)+j} = v_i + j - 1$ *for* $j = 1, \dots, m$
4. *Increment* i *and repeat 2–3 until* $i \geq \lceil T/m \rceil$
5. *Trim* u *so that only the first* T *remain if* T/m *is not an integer*

MATLAB Code for Moving Block Bootstrap

```
% Block size
m = 10;
% Loop over B bootstraps
for b=1:B
    % ceil(T/m) Uniform random numbers over 1...T-m+1
    u = ceil((T-m+1)*rand(ceil(T/m),1));
    u = bsxfun(@plus,u,0:m-1)';
    % Transform to col vector, and remove excess
    u = u(:); u = u(1:T);
    % y-star sample simulation
    yStar = y(u);
    % Mean of y-star
    muStar(b) = mean(yStar);
end
```

Circular Bootstrap

- Simple extension of MBB which assumes the data live on a circle so that $y_{T+1} = y_1$, $y_{T+2} = y_2$, etc.
- Has better finite sample properties since all data points get sampled with equal probability
- Only step 2 changes in a very small way

Algorithm (Circular Block Bootstrap)

1. *Initialize* $i = 1$
2. *Draw a uniform integer* v_i *on* $1, \ldots, T$
3. *Assign* $u_{(i-1)+j} = v_i + j - 1$ *for* $j = 1, \ldots, m$
4. *Increment* i *and repeat 2–3 until* $i \geq \lceil T/m \rceil$
5. *Trim* u *so that only the first* T *remain if* T/m *is not an integer*

MATLAB Code for Circular Block Bootstrap

UNIVERSITY OF OXFORD

```
% Block size
m = 10;
% Loop over B bootstraps
yRepl = [y;y];
for b=1:B
    % ceil(T/m) Uniform random numbers over 1...T-m+1
    u = ceil(T*rand(ceil(T/m),1));
    u = bsxfun(@plus,u,0:m-1)';
    % Transform to col vector, and remove excess
    u = u(:); u = u(1:T);
    % y-star sample simulation
    yStar = yRepl(u);
    % Mean of y-star
    muStar(b) = mean(yStar);
end
```

Stationary Bootstrap

- Differs form MBB and CBB in that the block size is no longer fixed
- Chooses an average block size of m rather than an exact block size
- Randomness in block size is worse when m is known, but helps if m may be suboptimal
- Block size is *exponentially distributed* with mean m

Algorithm (Stationary Bootstrap)

1. Draw u_1 uniform on $1, \ldots, T$
2. For $i = 2, \ldots, t$
   a. Draw a uniform v on $(0, 1)$
   b. If $v \geq 1/m$ $u_i = u_{i-1} + 1$
      i. If $u_i > T$, $u_i = u_i - T$
   c. If $v < 1/m$, draw u_i uniform on $1, \ldots, T$

MATLAB Code for Stationary Bootstrap

```
% Average block size
m = 10;
% Loop over B bootstraps
yRepl = [y;y];
u = zeros(T,1);
for b=1:B
    u(1) = ceil(T*rand);
    for t=2:T
        if rand<1/m
            u(t) = ceil(T*rand);
        else
            u(t) = u(t-1) + 1;
        end
    end
    % y-star sample simulation
    yStar = yRepl(u);
    % Mean of y-star
    muStar(b) = mean(yStar);
end
```

Comparing the Three TS Bootstraps

UNIVERSITY OF OXFORD

- MBB was the first
- CBB has simpler theoretical properties and usually requires fewer corrections to address “end effects”
- SB is theoretically worse than MBB and CBB, but is the most common choice in time-series econometrics
  - Theoretical optimality assumes that the the “optimal” block size is used
- Popularity of SB stems from difficulty in determining optimal m
  - More on this in a minute
- Random block size brings some robustness at the cost of extra variability

Bootstrapping Stationary AR(P)

- The stationary AR(P) model can be parametrically bootstraps
- Assume

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_P y_{t-P} + \epsilon_t$$

- Usual assumptions, including stationarity
- Can use a parametric bootstrap by estimating the residuals

$$\hat{\epsilon}_t = y_t - \hat{\phi}_1 y_{t-1} + \ldots + \hat{\phi}_P y_{t-P}$$

Algorithm (Stationary Autoregressive Bootstrap)

1. *Estimate the AR(P) and the residuals for* $t = P+1, \ldots, T$
2. *Recenter the residuals so that they have mean 0*

$$\tilde{\epsilon}_t = \hat{\epsilon}_t - \bar{\hat{\epsilon}}$$

3. *Draw u uniform from* $1, \ldots, T-P+1$ *and set* $y_1^\star = y_u$, $y_2^\star = y_{u+1}, \ldots, y_P^\star = y_{u+P+1}$
4. *Recursively simulate* $y_{P+1}^\star \ldots y_T^\star$ *using* $\tilde{\epsilon}$ *drawn using an i.i.d. bootstrap*

MATLAB Code for Stationary AR Bootstrap

```
phi = y(1:T-1)\y(2:T);
ehat = y(2:T)-phi*y(1:T-1);
yStar = zeros(T,1);
for i=1:B
    % Initialize to one of the original values
    yStar(1) = y(ceil(T*rand));
    % Indices for errors
    u = ceil((T-1)*rand(T,1));
    % Demean errors
    eStar = ehat(u) - mean(ehat(u))
    % Recursion to simulate AR
    for t=2:T
       yStar(t) = phi*yStar(t-1) + eStar(u(t));
    end
end
```

Data-based Block Length Selection

- Block size selection is crucial for good performance of block bootstraps
- Small block sizes are too close to i.i.d. while large block sizes are overly noisy
- Politis and White (2004) provide a data dependent lag length selection procedure
  - See also Patton, Politis, and White (2007) correction
- Code is available by searching the internet for "opt_block_length_REV_dec07"

How Data-based Block Length Selection Works

UNIVERSITY OF OXFORD

- Politis and White (2004) show for stationay bootstrap

$$B_{opt,SB} = \left(\frac{2G^2}{D_{SB}}\right) N^{1/3}$$

  - $G = \sum_{k=-\infty}^{\infty} |k| \gamma_k$ where γ_k is the autocovariance
  - $D_{SB} = 2g(0)^2$ where $g(w) = \sum_{s=-\infty}^{\infty} \gamma_s \cos(ws)$ is the spectral density function
- Need to estimate $\hat{G}$ and $\hat{D}_{SB}$ to estimate $\hat{B}_{opt,SB}$
  - $\hat{G} = \sum_{k=-M}^{M} \lambda\left(k/M\right) |k| \hat{\gamma}_k$,

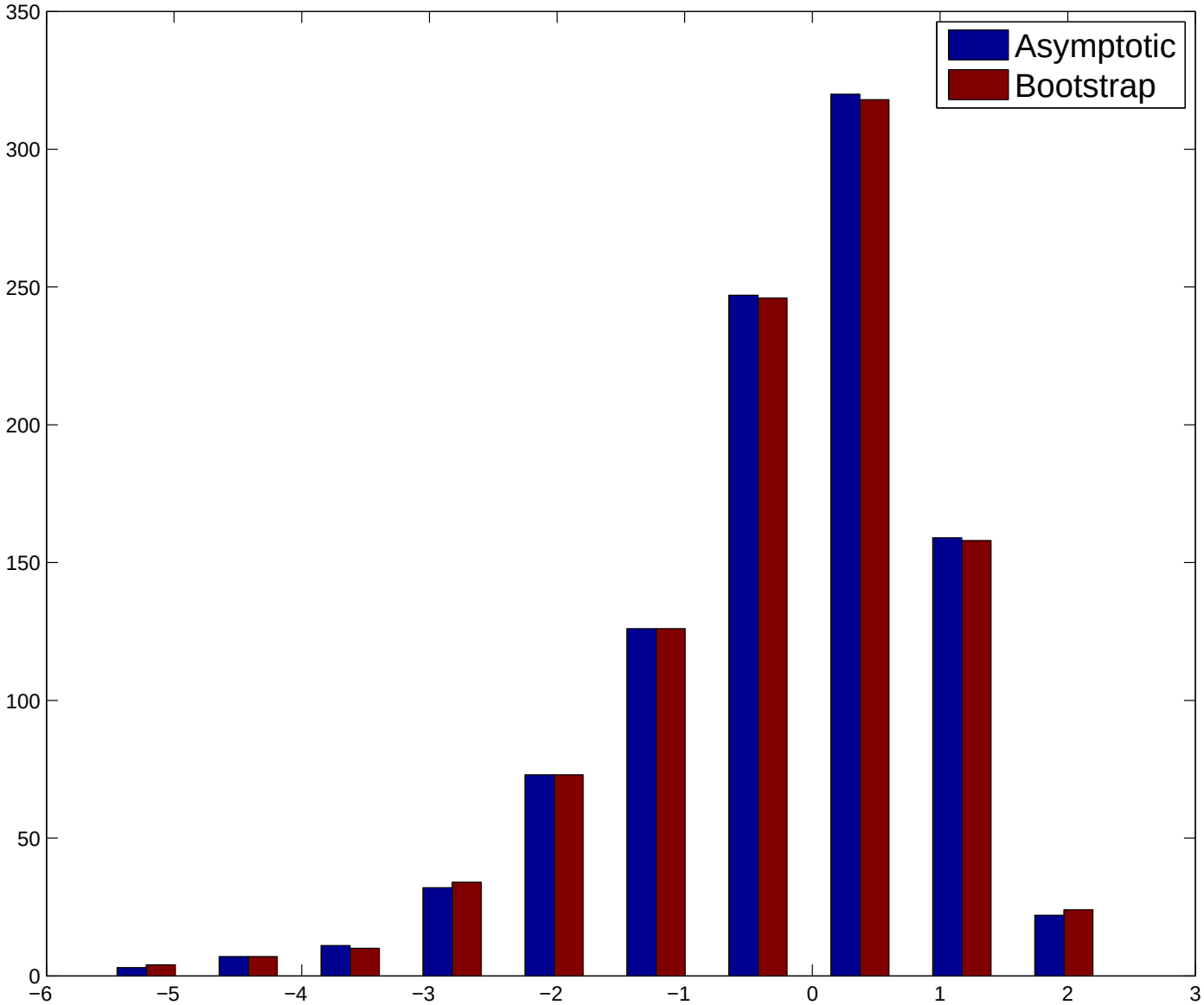
$$\lambda(s) = \begin{cases} 1 & \text{if } |s| \in [0, 1/2] \\ 2\left(1 - |s|\right) & \text{if } |s| \in [1/2, 1] \\ 0 & \text{otherwise} \end{cases}$$

  - $\hat{D}_{SB} = 2\hat{g}(0), \hat{g}(w) = \sum_{k=-M}^{M} \lambda\left(k/M\right) \hat{\gamma}_k \cos\left(wk\right)$
  - M is set to $2\hat{m}$
  - $\hat{m}$ is the smallest integer where if $\hat{\rho}_j > 2\sqrt{\log T/T}, j = m+1, \ldots, K_T$ where $K_T = 2\max\left(5, \sqrt{\log_{10}(T)}\right)$

Example 1: Mean Estimation for Log Normal

- $y_i \overset{\text{i.i.d.}}{\sim} LN(0, 1)$
- $n = 100$
- $B = 1000$ using i.i.d. bootstrap
- This is a check that the bootstrap works
- Also shows that bootstrap will not work miracles
- Performance of bootstrap is virtually identical to that of asymptotic theory
  - Gains to bootstrap are more difficult to achieve
  - Most useful property is in estimating standard error in hard to compute cases

Example 1: Mean Estimation for Log-Normal



Example 2: Bias in AR(1)

- Assume $y_t = \phi y_{t-1} + \epsilon_t$ where $\epsilon_t \overset{\text{i.i.d.}}{\sim} N(0, 1)$
- $\phi = 0.9$, $T = 50$
- Use parametric bootstrap
- Estimate bias using the different between bootstrap estimates and the actual estimate

	Direct	Debiased
$\hat{\phi}$	0.8711	0.8810
Var	0.0052	0.0044

- Reduced the bias by about 1/3
- Reduced variance (rare)

Technical Trading

- Technical trading is one form or predictive modeling
- It is mostly a graphical, rather than statistical tool
- Constructs rules based on price movements
- Rules, while often used graphically, can usually be written down in mathematical expressions
- This can be used to formally allow for testing for technical trading rules
  - Testing the rules is going to be the basis of the assignments this term
  - Using appropriate methodology for evaluation will be important

Data

- Daily DJIA for 12 months
- Use high, low and close
- Compute the rules, but focus on the visualization of the rule
- Rule implementation
  - Red dot is sell
  - Green dot is buy

Filter Rules

UNIVERSITY OF OXFORD

Definition ($x\%$ Buy Filter Rule)

A $x\%$ filter rule buys when price has increased by $x\%$ from the previous low, and liquidates when the price has declined $x\%$ from the high measured since the position was opened.

Definition ($x\%$ Sell Filter Rule)

A $x\%$ filter rule sells when price has declined by $x\%$ from the previous high, and liquidates when the price has increased $x\%$ from the low measured since the position was opened.

- These are a momentum rule
- If using both rules with the same percentage, will always have an long or short position, since after a decline of $x\%$, a short is opened, and after a rise of $x\%$ a long is opened

Filter Rules

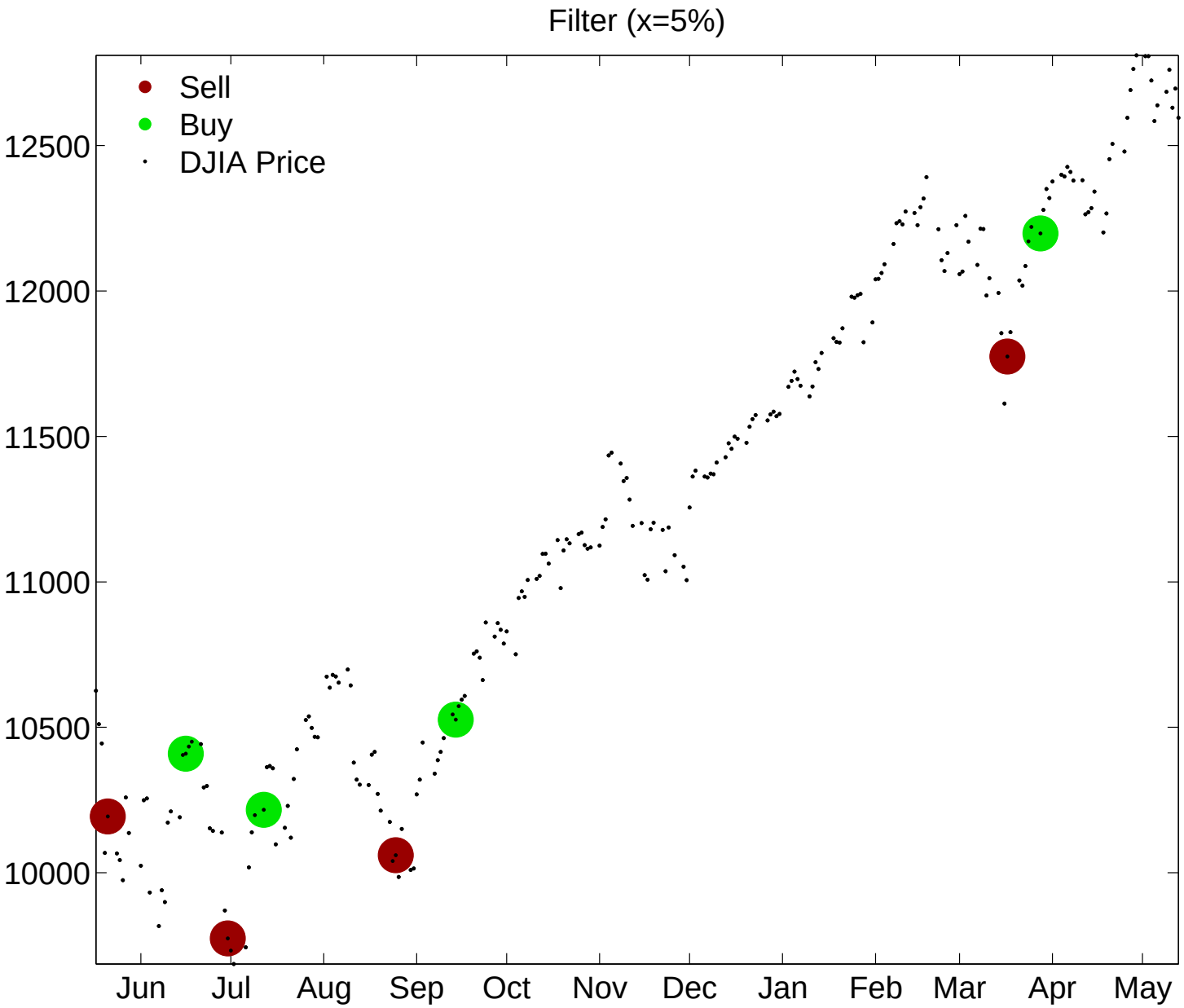
- A modified rule allows for periods where there is no long or short

Definition ($x\%/y\%$ Buy Filter Rule)

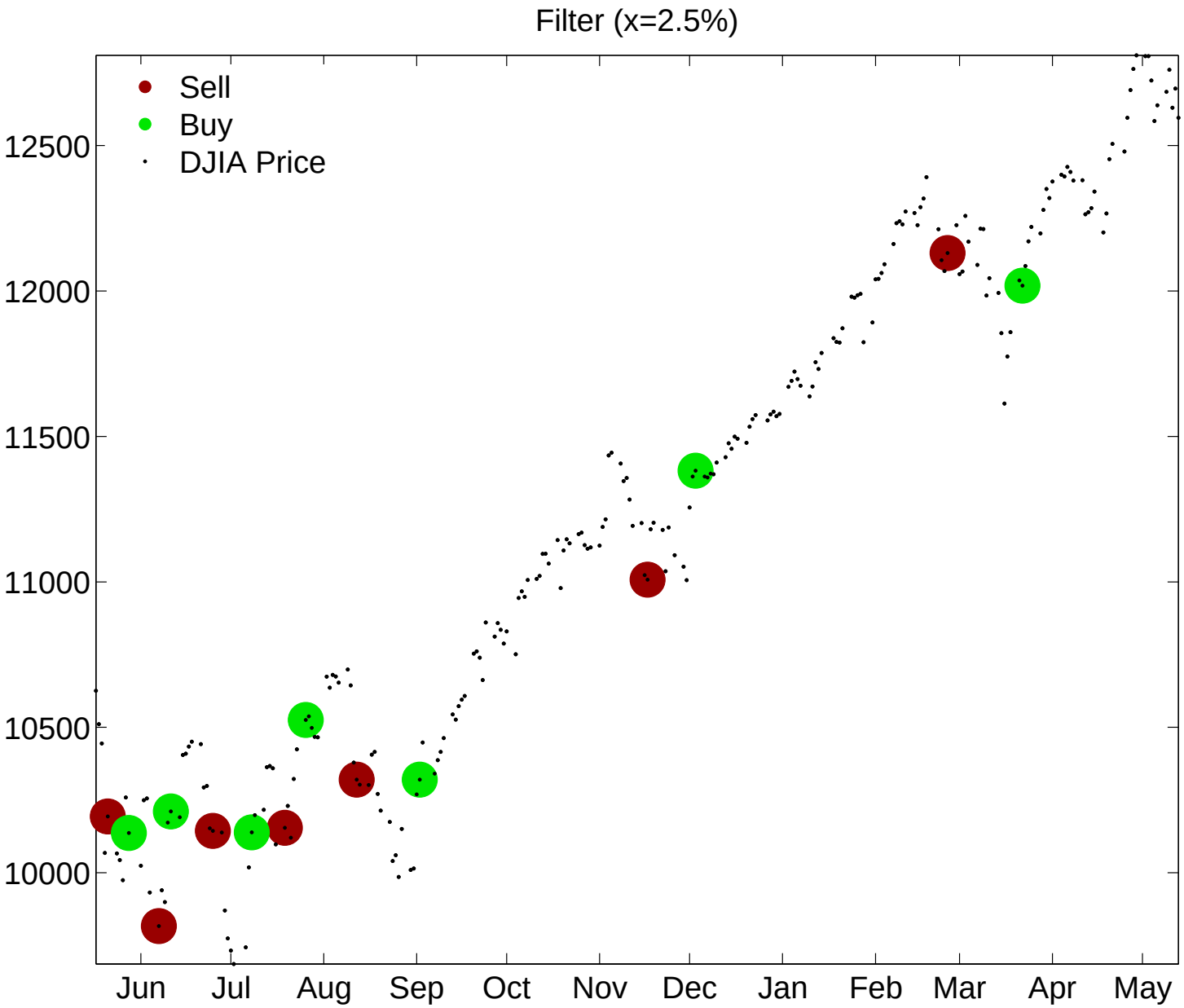
A $x\%$ filter rule buys when price has moved up by $x\%$ from the previous low, and liquidates when the price has declined $y\%$ from the high measured since the position was opened.

- The sell rule is similarly defined, only using the relative low
- $y \leq x$, and $y = x$ then reduces to previous rules
- Do not have to use both long and short rules

Filter Rules



Filter Rules



Moving-Average Oscillator

Definition (Moving-Average Oscillator)

The moving average oscillator requires two parameters, m and n, $n > m$,

$$MAO_t = m^{-1} \sum_{i=t-m+1}^{t} P_i - n^{-1} \sum_{i=t-n+1}^{t} P_i$$

- This is obviously the difference between an m period MA and a n period MA
- Momentum rule
- It is used as an indicator to buy when positive or sell when negative
  - Usually used to initiate a trade when it first crosses, not simply based on sign

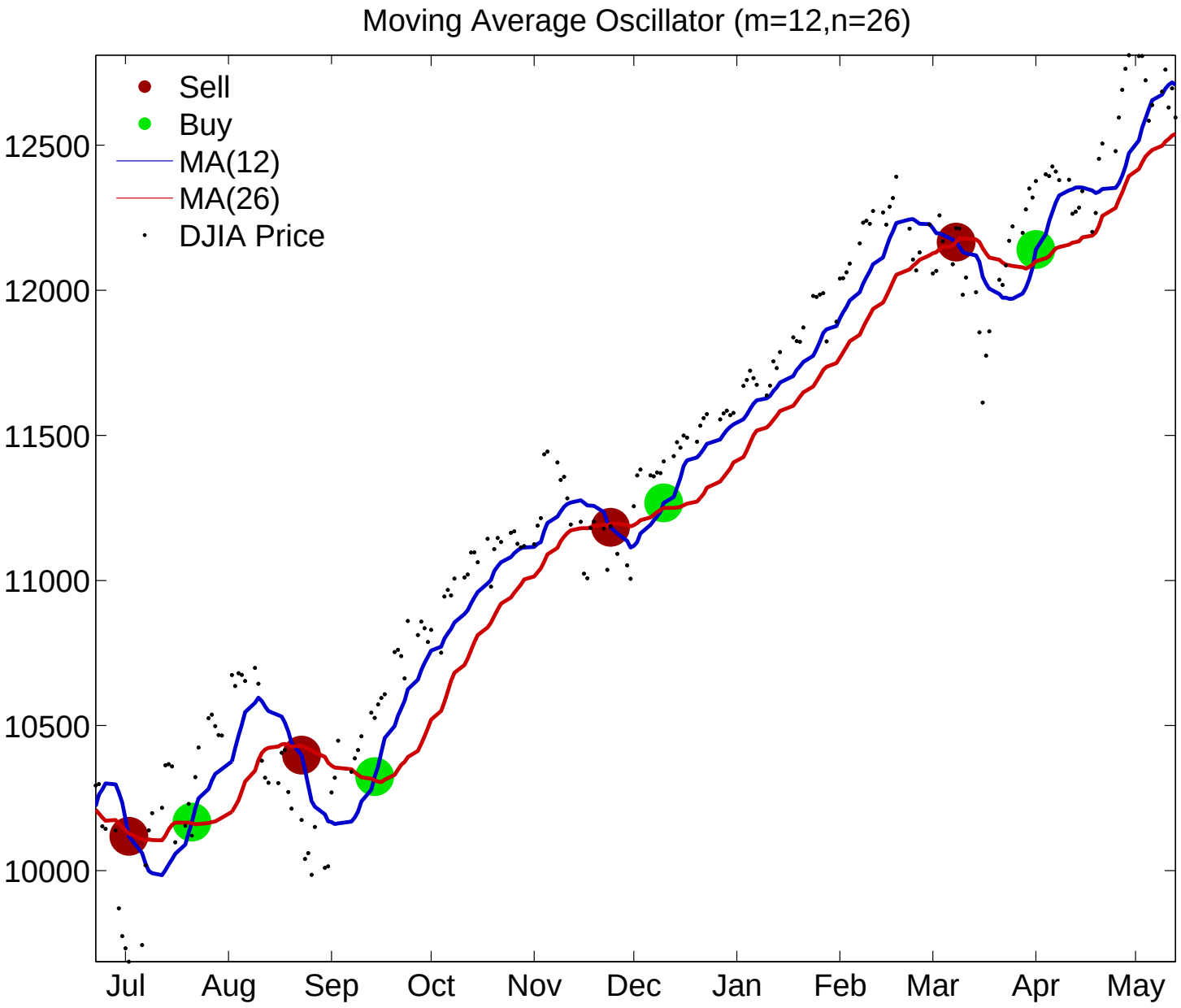
Moving-Average Oscillator

- MA_t is not enough to determine a buy rule, since the direction of the crossing matters
- Formally the buy and sell can be defined as the difference of MA_t

$$\begin{aligned} \text{Buy if} \quad & \operatorname{sgn}(MAO_t) - \operatorname{sgn}(MAO_{t-1}) = 2 \\ \text{Sell if} \quad & \operatorname{sgn}(MAO_t) - \operatorname{sgn}(MAO_{t-1}) = -2 \end{aligned}$$

- sgn is the signum function which returns $x/|x|$ for $x \neq 0$ and 0 for $x = 0$

Moving Average Oscillator



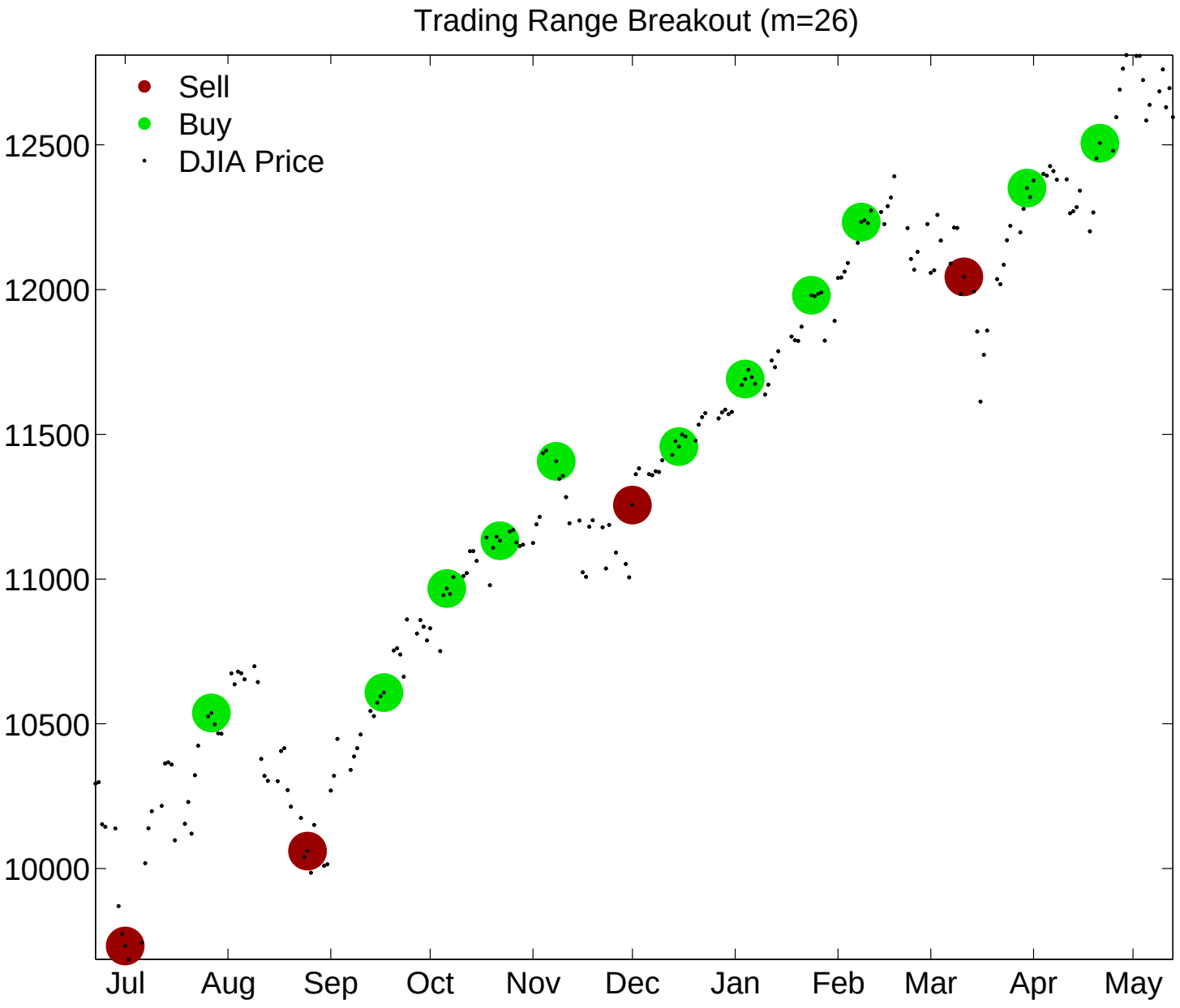
Definition (Trading Range Breakout)

The trading range break out is takes one parameter, m, and is defined

$$TRB_t = \left(P_t > \max\left(\{P_i\}_{i=t-m}^{t-1}\right)\right) - \left(P_t < \min\left(\{P_i\}_{i=t-m}^{t-1}\right)\right)$$

- Positive values (1) indicate that the price is above the m-period *moving maximum*, negative values -1 indicate that it is below the m-period *moving minimum*.
- Momentum rule
- Buy on positive signals, sell on negative signals
- If no signal, then takes the value 0

Trading Range Breakout



Definition (x% Channel Breakout)

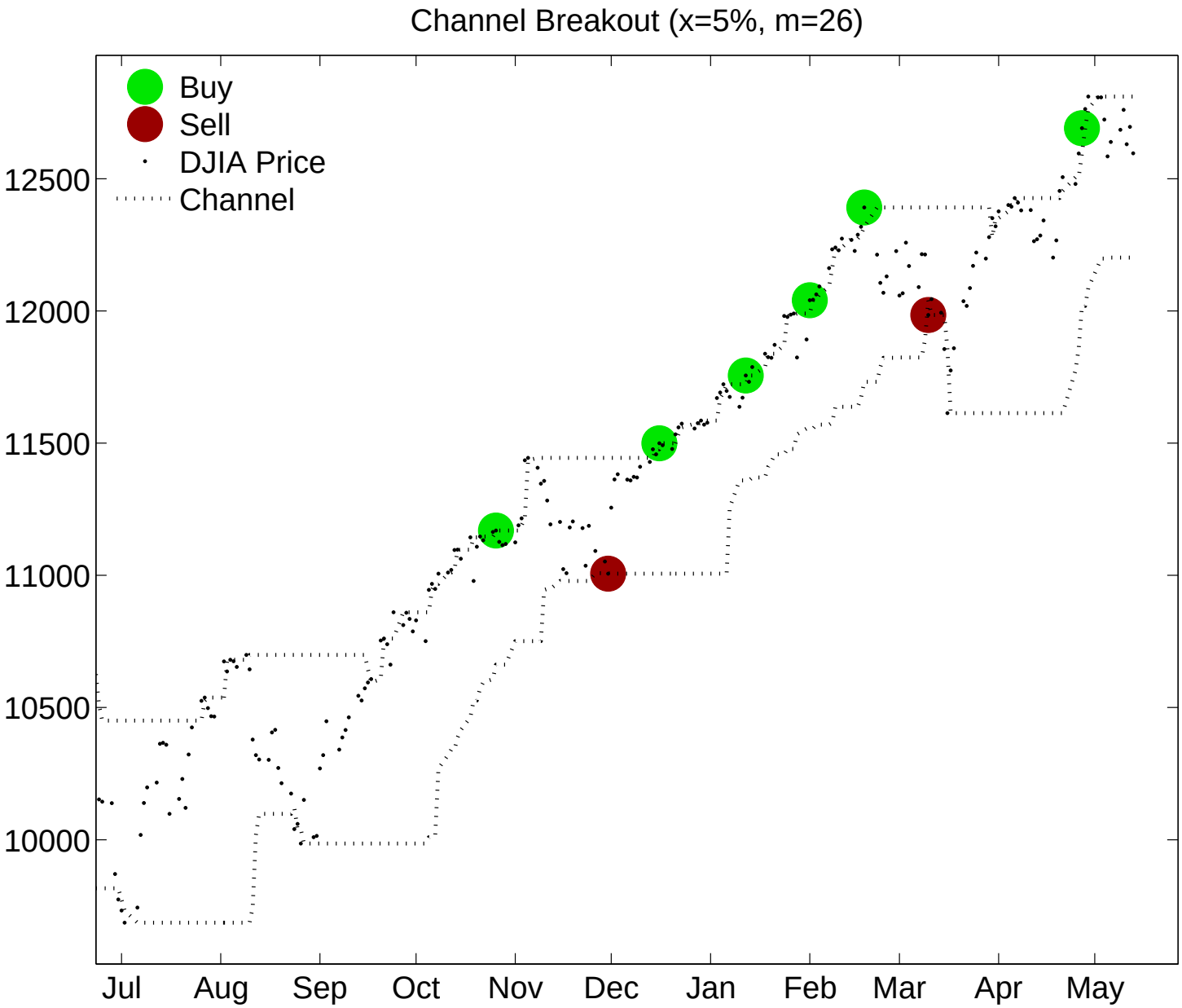
The $x\%$ channel breakout rule, using a m-day channel, is defined

$$\text{Buy if} \quad P_t > \max\left(\{P_i\}_{i=t-m}^{t-1}\right) \cap \frac{\max\left(\{P_i\}_{i=t-m}^{t-1}\right)}{\min\left(\{P_i\}_{i=t-m}^{t-1}\right)} < (1+x)$$

$$\text{Buy if} \quad P_t < \min\left(\{P_i\}_{i=t-m}^{t-1}\right) \cap \frac{\max\left(\{P_i\}_{i=t-m}^{t-1}\right)}{\min\left(\{P_i\}_{i=t-m}^{t-1}\right)} < (1+x)$$

- Momentum rule
- $x\%$ denotes the channel
- Modification of trading range breakout with second condition which may reduce sensitivity to volatility

Channel Range Breakout



Moving Average Convergence/Divergence (MACD)

UNIVERSITY OF OXFORD

Definition (Moving Average Convergence/Divergence (MACD))

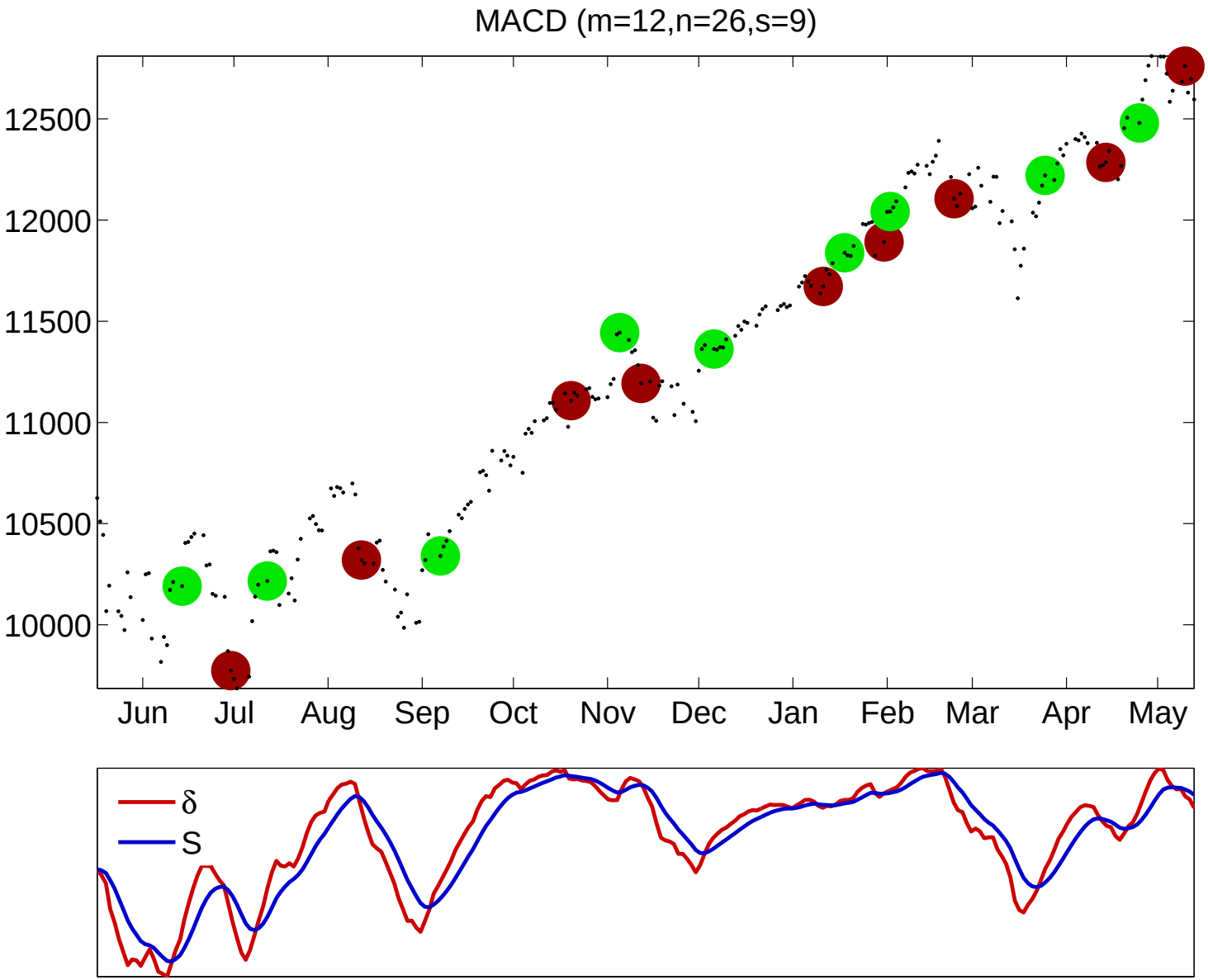
The moving-average convergence/divergence indicator takes three parameters, m, n and d, and is defined

$$\begin{aligned} \delta_t &= (1-\lambda_m)\sum_{i=0}^{\infty}\lambda_m^i P_{t-i} - (1-\lambda_n)\sum_{i=0}^{\infty}\lambda_n^i P_{t-i} \\ S_t &= (1-\lambda_d)\sum_{i=0}^{\infty}\lambda_d^i \delta_t \end{aligned}$$

- Pronounced MAK-D
- $\lambda_m = 1 - \frac{2}{m+1}$, $\lambda_n = 1 - \frac{2}{n+1}$,$\lambda_d = 1 - \frac{2}{d+1}$
- S_t is the signal line
- Plot often has δ and S, and a histogram to indicate the difference $\delta_t - S_t$
- Difference is used to predict trends

$$\begin{aligned} &\text{Buy if} && \operatorname{sgn}(\delta_t - S_t) - \operatorname{sgn}\left(\delta_{t-1} - S_{t-1}\right) = 2 \\ &\text{Sell if} && \operatorname{sgn}(\delta_t - S_t) - \operatorname{sgn}\left(\delta_{t-1} - S_{t-1}\right) = -2 \end{aligned}$$

Moving Average Convergence/Divergence



Relative Strength Indicator

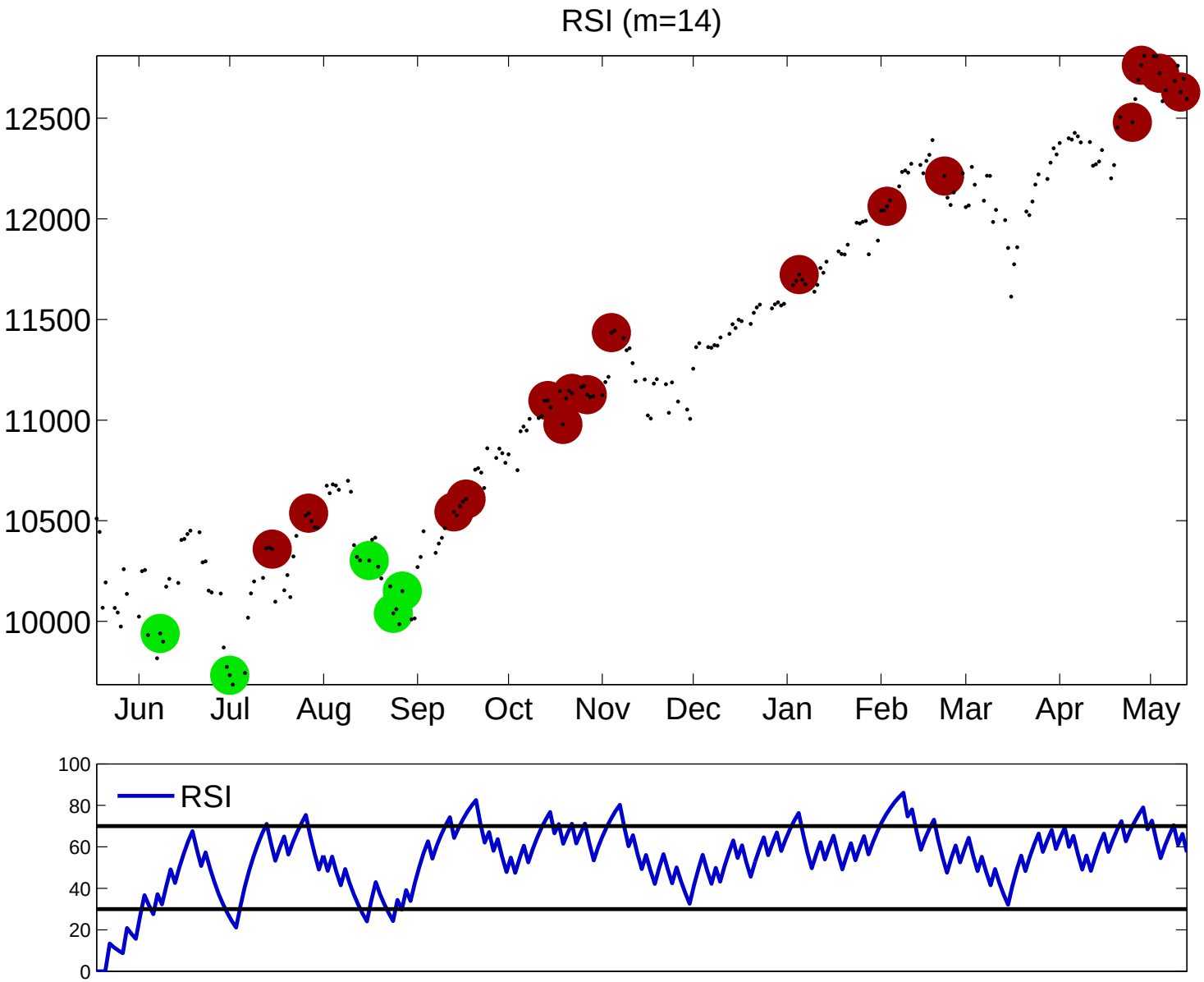
Definition (Relative Strength Indicator)

The relative strength indicator takes one parameter m and is defined as

$$RSI = 100 - \frac{100}{1 + \frac{\sum_{i=0}^{\infty} \lambda^i I_{\left[\left(P_{t-i} - P_{t-i-1}\right) > 0\right]}}{\sum_{i=0}^{\infty} \lambda^i I_{\left[\left(P_{t-i} - P_{t-i-1}\right) < 0\right]}}}, \quad \lambda = 1 - \frac{2}{m+1}$$

- The core of the indicator are two EWMAs
- Each EWMA is based on indicator variables or positive (top) or negative (bottom) returns
- If all positive, then indicator will equal 100, if all negative, indicator will equal 0
- EWMA can be replaced with MA
- Buy signals are indicated if RSI is *below* some threshold (e.g. 30), sell if *above* a different threshold (e.g. 70)
- RSI is a reversal rule

Relative Strength Indicator (Reversal)



Stochastic Oscillator

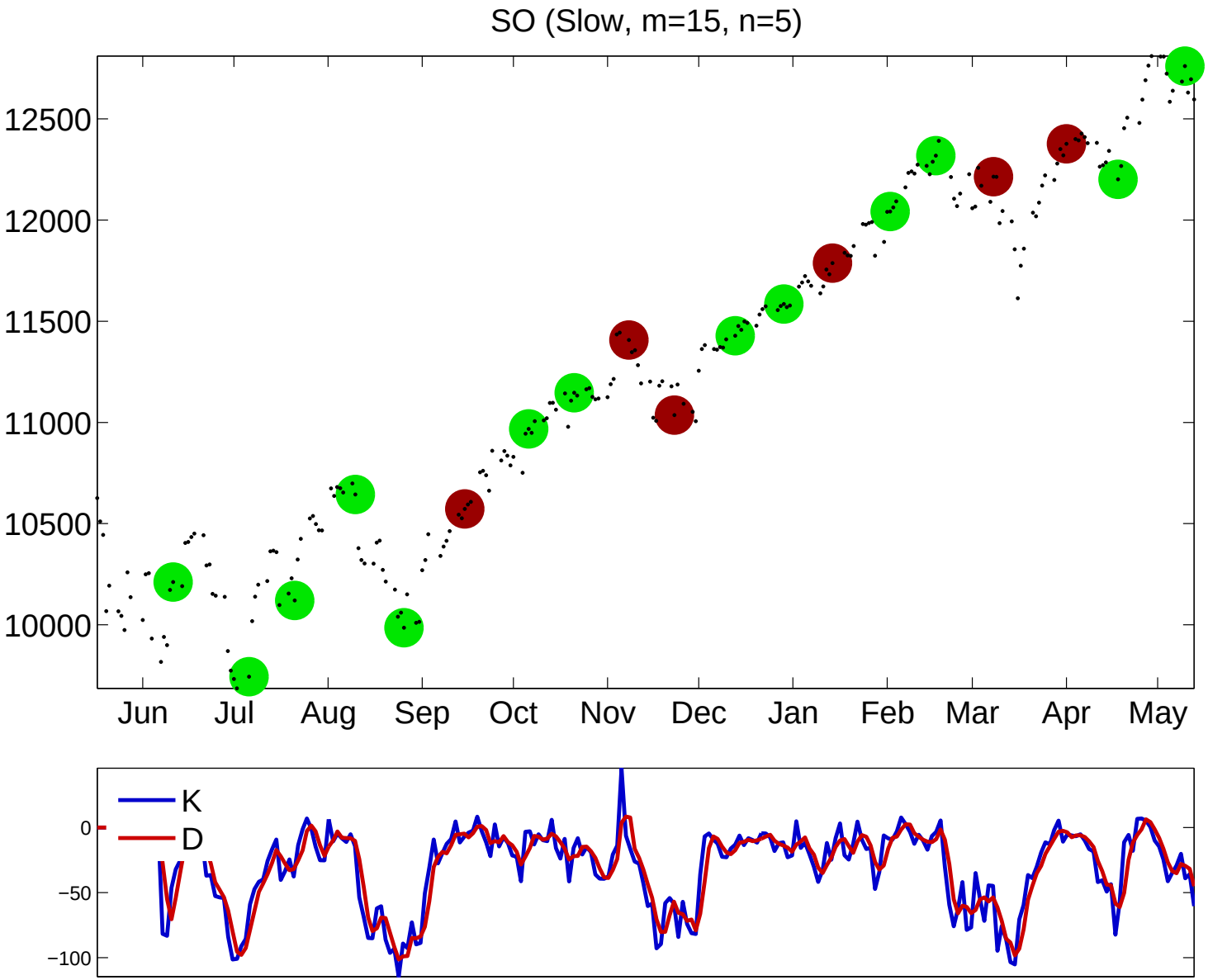
Definition (Stochastic Oscillator)

A stochastic oscillator takes two parameters m and n and is defined as

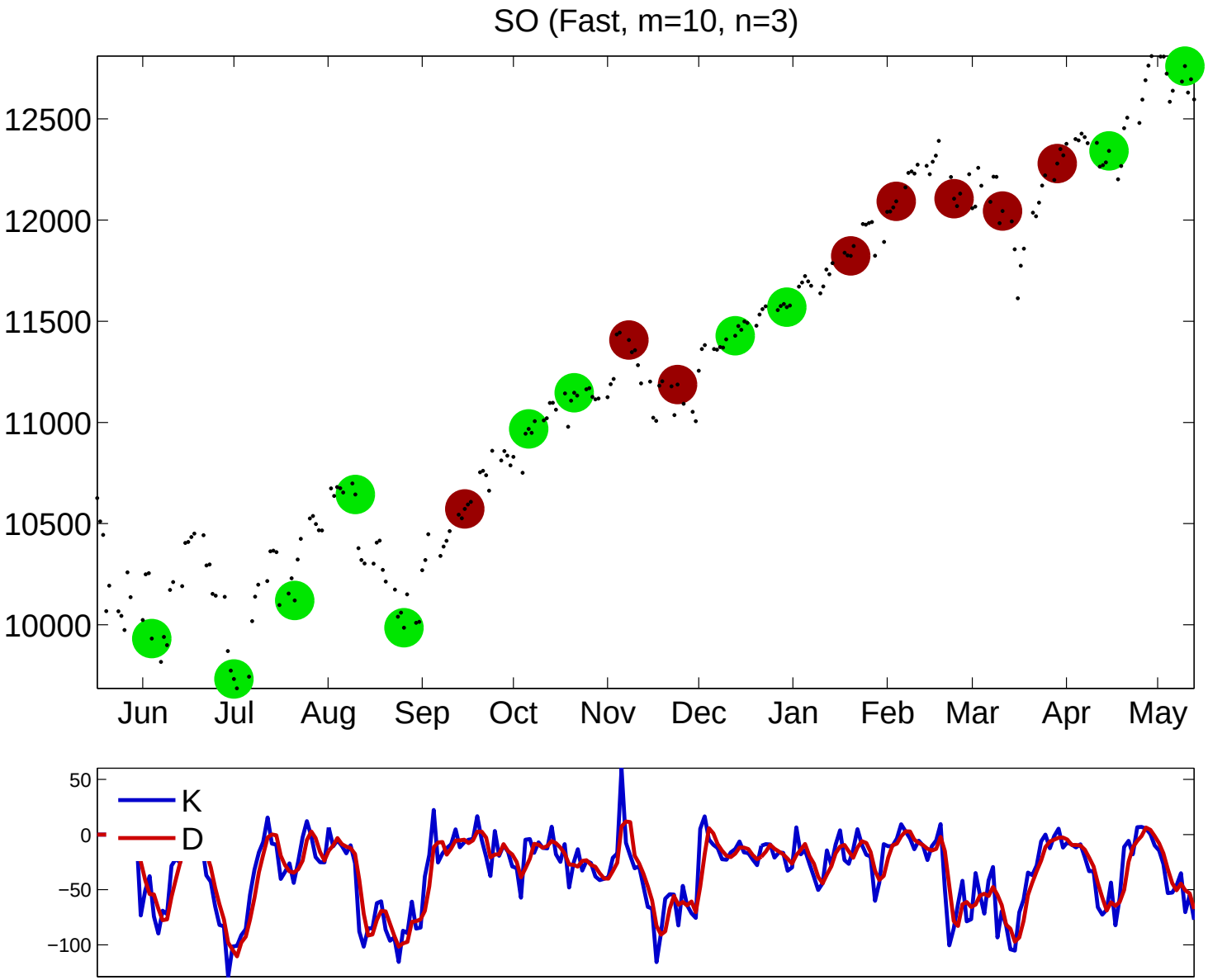
$$\begin{aligned} \%K_t &= 100 \times \frac{P_t - \min\left(\{P_i\}_{i=t-m}^{t-1}\right)}{\max\left(\{P_i\}_{i=t-m}^{t-1}\right) - \min\left(\{P_i\}_{i=t-m}^{t-1}\right)} \\ \%D_t &= \frac{1}{n}\sum_{i=1}^{n} \%K_{t-i+1} \end{aligned}$$

- Trading rules are based on intersections of the lines *and* the direction of of the intersection
- If $\%K_{t-1} < \%D_{t-1}$ and $\%K_t > \%D_t$, then a buy signal is indicated
- If $\%K_{t-1} > \%D_{t-1}$ and $\%K_t < \%D_t$, then a sell signal is indicated
- Often implemented using *fast* and *slow* periods, with feedback between the two

Stochastic Oscillator



Stochastic Oscillator



Bollinger Band

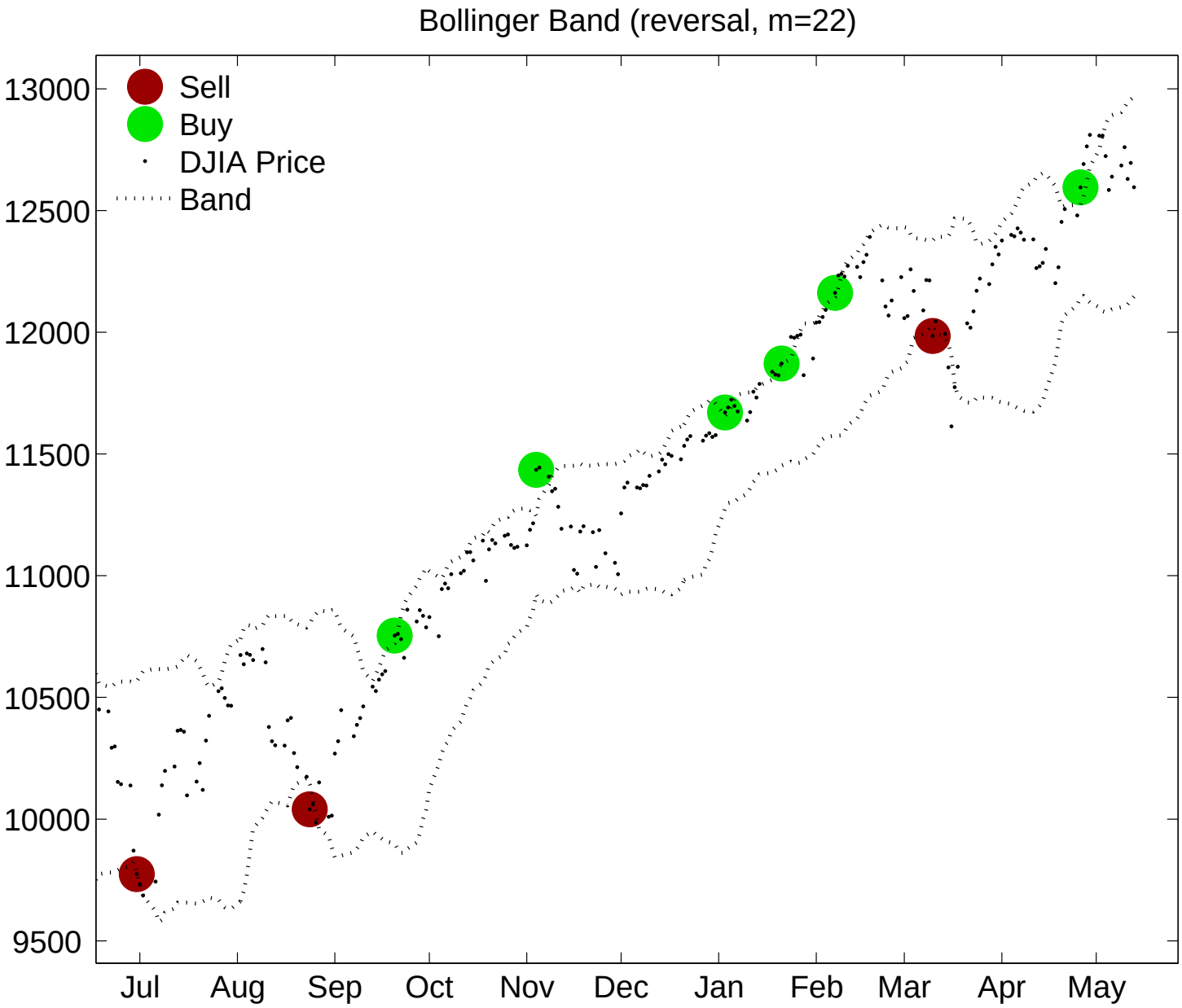
Definition (Bollinger Bands)

Bollinger bands plot the m-day moving average and the MA plus/minus 2 times the m-day moving standard deviation, where the moving averages are defined

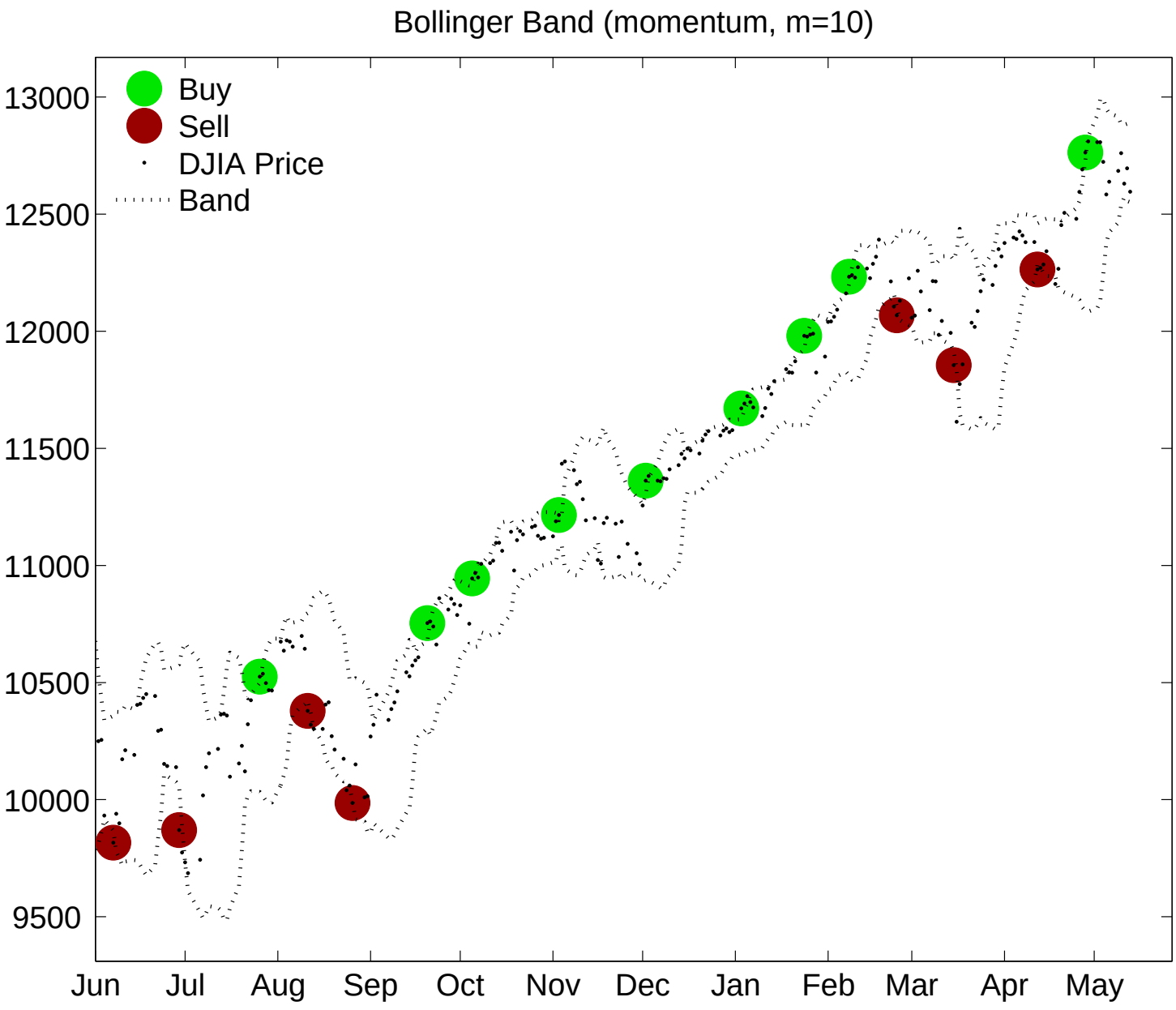
$$MA_t = m^{-1}\sum_{i=1}^{m} P_{t-i+1}, \sigma_t = \sqrt{m^{-1}\sum_{i=1}^{m}\left(\frac{\left(P_{t-i+1}-P_{t-i}\right)}{P_{t-i}}\right)^2}$$

- Rules can be based on prices leaving the bands, and possibly then crossing of the moving average
- For example, buy when price hit bottom (reversal) and then sell when it hits the MA
- Alternatively buy when it hits the top (strong upward trend)

Bollinger Band



Bollinger Band



A Simple Momentum Rule

- Momentum is a common strategy
- Can construct a momentum rule as

$$S_t = \begin{cases} 1 & \text{if } P_t > P_{t-d} \\ 0 & \text{if } P_t \leq P_{t-d} \end{cases}$$

- Technically (trivial) moving average rule with d-day delay filter

On-Balance Volume

Definition (On-Balance Volume)

On-Balance Volume (OBV) plots the difference between moving averages of signed daily volume, defined

$$OBV_t = \sum_{s=1}^{t} VOL_s D_s$$

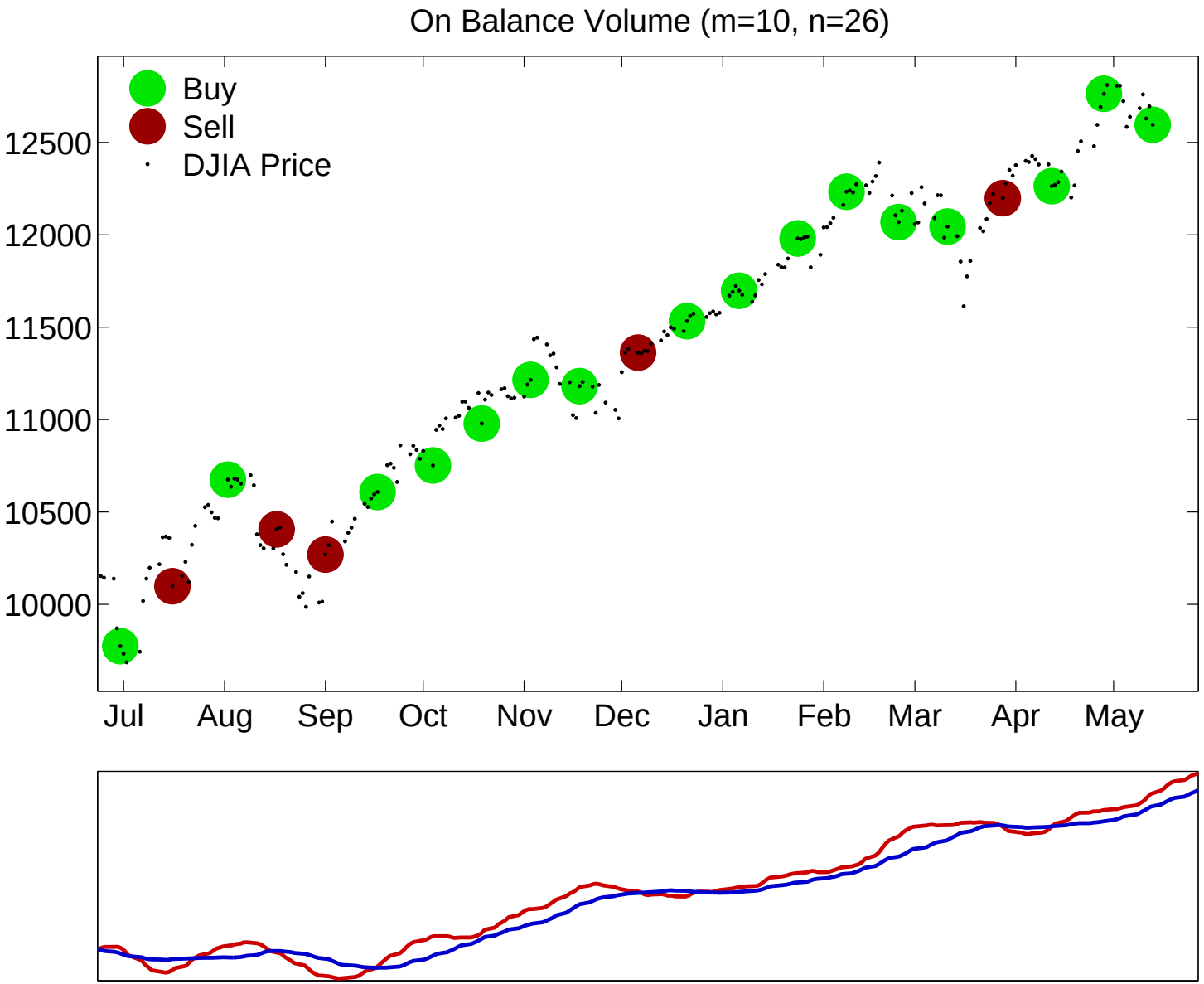
where VOL_s is the volume in period s, D_s is a dummy which is 1 if $P_t > P_{t-1}$ and -1 otherwise, and the trading signal is

$$S_t = \begin{cases} 1 & MA_{m,t}^{OBV} > MA_{n,t}^{OBV} \\ 0 & MA_{m,t}^{OBV} \leq MA_{n,t} \end{cases}$$

where $MA_{q,t}^{OBV} = q^{-1} \sum_{i=1}^{q} OBV_{t-i-1}$, $q = m, n$, $m < n$.

- Most rules make use of price signals
- OBV mixes volume information with indicator variable

On-Balance Volume



Additional Filters

- Many ways rules can be modified
- MAs and EWMAs can be swapped
- Can use a d-day delay filter to stagger execution of trade from signal
- Can use $b\%$-band with some filters to reduce frequency of execution
  - Requires the recent price (or fast signal) to be $b\%$ above the band (or slow signal)
  - Relevant for most rules
  - Examples
    - Moving-Average Oscillator: Requires fast MA to be larger than $1+b$ times slow for a buy signal, and smaller than $1-b$ for a sell signal
    - Trading Range Breakout/Channel Breakout: Use $1+b$ times max and $1-b$ times min
- Can use k-day holding period, so that positions are held for k-days and other signal are ignored

From Technical Indicators to Trading Rules

UNIVERSITY OF OXFORD

- Most technical rules are interpreted as buy, neutral or sell – 1, 0 or -1
- Essentially applies a step function to the trading signal
- Can use a other continuous, monotonic increasing functions, although not clear which ones
- One options is to run a regression

$$r_{t+1} = \beta_0 + \beta_1 S_t + \epsilon_t$$

- S_t is a signal is computed using information up-to and including t
  - Can be discrete or continuous
- Maps to an expected return, which can then be used in Sharpe-optimization

Combining Multiple Technical Indicators

- Technical trading rules can be combined
- Not obvious how to combine when discrete
- Method 1: Majority vote
  - Count number of rules with signs 1, 0 or -1
- Method 2: Aggregation
  - Compute sum of indicators divided by number of indicators

$$\tilde{S}_t = \frac{\sum_{i=1}^{k} S_{k,t}}{k}$$

  and go long/short $\tilde{S}_t$
  - Bound by 100% long and 100% short

Evaluating the Rules

- Obvious strategy it to look at returns, conditional on signal
- Important to have a benchmark model
  - Often buy and hold, or some other much less dynamic strategy
- Obvious test is t-statistic of difference in mean return between the active strategy and the benchmark
- Can also examine predictability for other aspects of distribution
  - Volatility
  - Large declines

Brock, Lakonishok and LeBaron

- One of the first systematically test trading rules
- Focused on two rules:
  - Moving Average Oscillator
  - Trading Range Breakout
- (Controversially) documented evidence of excess returns to technical trading rules
- Returns were large enough to cover transaction costs

Moving Average Oscillator

- Moving Average Oscillators implemented for
  - $m = 1, n = 50$
  - $m = 1, n = 150$
  - $m = 5, n = 150$
  - $m = 1, n = 200$
  - $m = 2, n = 200$
- Use both the standard rule and one with a 1%-band filter
- Standard is implemented by taking the position and holding for 10 days, ignoring all other signals
- $b\%$-band version:
  - Requires an exceedence by 1% of the slow MA, but no crossing

$$\text{Buy if } \left(\frac{MA_t}{n^{-1}\sum_{i=t-n+1}^{t} P_i}\right) > \frac{b}{100}, \text{ Sell if } \left(\frac{MA_t}{n^{-1}\sum_{i=t-n+1}^{t} P_i}\right) < -\frac{b}{100}$$

  - If $b > 0$ then some days may have no signal
  - If $b = 0$ then all days are buys or sells

Trading Range Breakout

- Trading range breakout is implemented for
  - $m = 50$
  - $m = 100$
  - $m = 150$
- Implemented using the standard and with a 1% band
- $b\%$ band version is

$$\begin{aligned} TRB_t = & \left(P_t > \left(1 + \frac{b}{100} \right) \max \left(\{P_i\}_{i=t-m}^{t-1} \right) \right) \\ & - \left(P_t < \left(1 - \frac{b}{100} \right) \min \left(\{P_i\}_{i=t-m}^{t-1} \right) \right) \end{aligned}$$

Empirical Application

- A total of 26 rules are created
  - MAO: 5 $(m, n)\times$ 2 (Fixed or Variable Window) $\times 2$ $(b = 0, .01)$
  - TRB: 3 (m) $\times$ 2 $(b = 0, .01)$
- DJIA from 1897 until 1986
- Main result is that there appears to be predictability using these rules
- Strongest results were for the fixed windows MAO with $m = 1$, $n = 200$ and $b = .01$
- TRB with $m = 150$ and $b = .01$ also had a strong result
- Report
  - Number of buy and sell signals
  - Mean return during buy and sell signals
  - Probability of positive return for buy and sell signals
  - Mean return of a portfolio which both buys and sells

the difference of the mean buy and mean sell from the unconditional 1-day mean, and buy-sell from zero. "Buy > 0" and "Sell > 0" are the fraction of buy and sell returns greater than zero. The last row reports averages across all 10 rules. Results for subperiods are given in Panel B.

Panel A: Full Sample

Period	Test	N(Buy)	N(Sell)	Buy	Sell	Buy > 0	Sell > 0	Buy-Sell
1897–1986	(1, 50, 0)	14240	10531	0.00047	−0.00027	0.5387	0.4972	0.00075
				(2.68473)	(−3.54645)			(5.39746)
	(1, 50, 0.01)	11671	8114	0.00062	−0.00032	0.5428	0.4942	0.00094
				(3.73161)	(−3.56230)			(6.04189)
	(1, 150, 0)	14866	9806	0.00040	−0.00022	0.5373	0.4962	0.00062
				(2.04927)	(−3.01836)			(4.39500)
	(1, 150, 0.01)	13556	8534	0.00042	−0.00027	0.5402	0.4943	0.00070
				(2.20929)	(−3.28154)			(4.68162)
	(5, 150, 0)	14858	9814	0.00037	−0.00017	0.5368	0.4970	0.00053
				(1.74706)	(−2.61793)			(3.78784)
	(5, 150, 0.01)	13491	8523	0.00040	−0.00021	0.5382	0.4942	0.00061
				(1.97876)	(−2.78835)			(4.05457)
	(1, 200, 0)	15182	9440	0.00039	−0.00024	0.5358	0.4962	0.00062
				(1.93865)	(−3.12526)			(4.40125)
	(1, 200, 0.01)	14105	8450	0.00040	−0.00030	0.5384	0.4924	0.00070
				(2.01907)	(−3.48278)			(4.73045)
	(2, 200, 0)	15194	9428	0.00038	−0.00023	0.5351	0.4971	0.00060
				(1.87057)	(−3.03587)			(4.26535)
	(2, 200, 0.01)	14090	8442	0.00038	−0.00024	0.5368	0.4949	0.00062
				(1.81771)	(−3.03843)			(4.16935)
	Average			0.00042	−0.00025			0.00067

Panel B: Subperiods

generate a signal. N(Buy) and N(Sell) are the number of buy and sell signals reported during the sample. Numbers in parentheses are standard t-ratios testing the difference of the mean buy and mean sell from the unconditional 1-day mean, and buy-sell from zero. "Buy > 0" and "Sell > 0" are the fraction of buy and sell returns greater than zero. The last row reports averages across all 10 rules.

Test	N(Buy)	N(Sell)	Buy	Sell	Buy > 0	Sell > 0	Buy-Sell
(1, 50, 0)	340	344	0.0029	−0.0044	0.5882	0.4622	0.0072
			(0.5796)	(−3.0021)			(2.6955)
(1, 50, 0.01)	313	316	0.0052	−0.0046	0.6230	0.4589	0.0098
			(1.6809)	(−3.0096)			(3.5168)
(1, 150, 0)	157	188	0.0066	−0.0013	0.5987	0.5691	0.0079
			(1.7090)	(−1.1127)			(2.0789)
(1, 150, 0.01)	170	161	0.0071	−0.0039	0.6529	0.5528	0.0110
			(1.9321)	(−1.9759)			(2.8534)
(5, 150, 0)	133	140	0.0074	−0.0006	0.6241	0.5786	0.0080
			(1.8397)	(−0.7466)			(1.8875)
(5, 150, 0.01)	127	125	0.0062	−0.0033	0.6614	0.5520	0.0095
			(1.4151)	(−1.5536)			(2.1518)
(1, 200, 0)	114	156	0.0050	−0.0019	0.6228	0.5513	0.0069
			(0.9862)	(−1.2316)			(1.5913)
(1, 200, 0.01)	130	127	0.0058	−0.0077	0.6385	0.4724	0.0135
			(1.2855)	(−2.9452)			(3.0740)
(2, 200, 0)	109	140	0.0050	−0.0035	0.6330	0.5500	0.0086
			(0.9690)	(−1.7164)			(1.9092)
(2, 200, 0.01)	117	116	0.0018	−0.0088	0.5556	0.4397	0.0106
			(0.0377)	(−3.1449)			(2.3069)
Average			0.0053	−0.0040			0.0093

generate a signal. "N(Buy)" and "N(Sell)" are the number of buy and sell signals reported during the sample. Numbers in parentheses are standard t-ratios testing the difference of the mean buy and mean sell from the unconditional 1-day mean, and buy-sell from zero. "Buy > 0" and "Sell > 0" are the fraction of buy and sell returns greater than zero. The last row reports averages across all 6 rules.

Test	N(Buy)	N(Sell)	Buy	Sell	Buy > 0	Sell > 0	Buy-Sell
(1, 50, 0)	722	415	0.0050	0.0000	0.5803	0.5422	0.0049
			(2.1931)	(−0.9020)			(2.2801)
(1, 50, 0.01)	248	252	0.0082	−0.0008	0.6290	0.5397	0.0090
			(2.7853)	(−1.0937)			(2.8812)
(1, 150, 0)	512	214	0.0046	−0.0030	0.5762	0.4953	0.0076
			(1.7221)	(−1.8814)			(2.6723)
(1, 150, 0.01)	159	142	0.0086	−0.0035	0.6478	0.4789	0.0120
			(2.4023)	(−1.7015)			(2.9728)
(1, 200, 0)	466	182	0.0043	−0.0023	0.5794	0.5000	0.0067
			(1.4959)	(−1.4912)			(2.1732)
(1, 200, 0.01)	146	124	0.0072	−0.0047	0.6164	0.4677	0.0119
			(1.8551)	(−1.9795)			(2.7846)
Average			0.0063	−0.0024			0.0087

The Standard Forecasting Model

- Standard forecasts are also popular for predicting economic variables
- Generically expressed

$$y_{t+1} = \beta_0 + \mathbf{x}_t \boldsymbol{\beta} + \epsilon_{t+1}$$

- $\mathbf{x}_t$ is a 1 by k vector of predictors ($k = 1$ is common)
- Includes both exogenous regressors such as the term or default premium and also autoregressive models
- Forecasts are $\hat{y}_{t+1|t}$

The forecast combination problem

- Two level of aggregation in the combination problem

1. Summarize individual forecasters' private information in point forecasts $\hat{y}_{t+h,i|t}$
   - Highlights that "inputs" are not the usual explanatory variables, but forecasts
2. Aggregate individual forecasts into consensus measure $C\left(\mathbf{y}_{t+h|t}, \mathbf{w}_{t+h|t}\right)$

- Obvious competitor is the "super-model" or "kitchen-sink" – a model built using all information in each forecasters information set
- Aggregation should increase the bias in the forecast relative to SM but may reduce the variance
- Similar to other model selection procedures in this regard

Why not use the "Super Model"

- Could consider pooling information sets

$$\mathcal{F}_t^c = \cup_{i=1}^n \mathcal{F}_{t,i}$$

- Would contain all information available to all forecasters
- Could construct consensus directly $C\left(\mathcal{F}_t^c; \boldsymbol{\theta}_{t+h|t}\right)$
- Some reasons why this may not work
  - Some information in individuals information sets may be qualitative, and so expensive to quantitatively share
  - Combined information sets may have a very high dimension, so that finding the best super model may be hard
    - Potential for lots of estimation error
- Classic bias-variance trade-off is main reason to consider forecasts combinations over a super model
  - Higher bias, lower variance

Linear Combination under MSE Loss

- Models can be combined in many ways for virtually any loss function
- Most standard problem is for MSE loss using only linear combinations
- I will suppress time subscripts when it is clear that it is $t+h|t$
- Linear combination problem is

$$\min_{\mathbf{w}} \mathrm{E}\left[e^2\right] = \mathrm{E}\left[\left(y_{t+h} - \mathbf{w}'\hat{\mathbf{y}}\right)^2\right]$$

- Requires information about first 2 moments of he joint distribution of the realization y_{t+h} and the time-t forecasts $\hat{\mathbf{y}}$

$$\begin{bmatrix} y_{t+h|t} \\ \hat{\mathbf{y}} \end{bmatrix} \sim F\left(\begin{bmatrix} \mu_y \\ \boldsymbol{\mu}_{\hat{\mathbf{y}}} \end{bmatrix}, \begin{bmatrix} \sigma_{yy} & \boldsymbol{\Sigma}'_{y\hat{\mathbf{y}}} \\ \boldsymbol{\Sigma}_{y\hat{\mathbf{y}}} & \boldsymbol{\Sigma}_{\hat{\mathbf{y}}\hat{\mathbf{y}}} \end{bmatrix}\right)$$

Linear Combination under MSE Loss

- The first order condition for this problem is

$$\frac{\partial \mathrm{E}\left[e^2\right]}{\partial \mathbf{w}} = -\mu_y \boldsymbol{\mu}_{\hat{\mathbf{y}}} + \boldsymbol{\mu}_{\hat{\mathbf{y}}} \boldsymbol{\mu}_{\hat{\mathbf{y}}}' \mathbf{w} + \boldsymbol{\Sigma}_{\hat{\mathbf{y}}\hat{\mathbf{y}}} \mathbf{w} - \boldsymbol{\Sigma}_{y\hat{\mathbf{y}}} = \mathbf{0}$$

- The solution to this problem is

$$\mathbf{w}^\star = \left(\boldsymbol{\mu}_{\hat{\mathbf{y}}} \boldsymbol{\mu}_{\hat{\mathbf{y}}}' + \boldsymbol{\Sigma}_{\hat{\mathbf{y}}\hat{\mathbf{y}}}\right)^{-1} \left(\boldsymbol{\Sigma}_{y\hat{\mathbf{y}}} + \mu_y \boldsymbol{\mu}_{\hat{\mathbf{y}}}\right)$$

- Similar to the solution to the OLS problem, only with extra terms since the forecasts may not have the same conditional mean

Linear Combination under MSE Loss

- Can remove the conditional mean if the combination is allowed to include a constant, w_c

$$\begin{aligned} w_c &= \mu_y - \mathbf{w}^\star \boldsymbol{\mu}_{\hat{\mathbf{y}}} \\ \mathbf{w}^\star &= \boldsymbol{\Sigma}_{\hat{\mathbf{y}}\hat{\mathbf{y}}}^{-1} \boldsymbol{\Sigma}_{y\hat{\mathbf{y}}} \end{aligned}$$

- These are identical to the OLS where w_c is the intercept and $\mathbf{w}^*$ are the slope coefficients
- The role of w_c is the correct for any biases so that the squared bias term in the MSE is 0

$$\mathrm{MSE}\left[e\right] = \mathrm{B}\left[e\right]^2 + \mathrm{V}\left[e\right]$$

Understanding the Diversification Gains

- Simple setup

$$e_1 \sim F_1\left(0, \sigma_1^2\right), \; e_2 \sim F_2\left(0, \sigma_2^2\right), \; \text{Corr}\left[e_1, e_2\right] = \rho, \; \text{Cov}\left[e_1 e_2\right] = \sigma_{12}$$

- Assume $\sigma_2^2 \leq \sigma_1^2$
- Assume weights sum to 1 so that $w_1 = 1 - w_2$ (Will suppress the subscript and simply write w)
- Forecast error is then

$$y - w\hat{y}_1 - (1 - w)\,\hat{y}_2$$

- Error is given by

$$e^c = we_1 + (1 - w)\,e_2$$

- Forecast has mean 0 and variance

$$w^2\sigma_1^2 + (1 - w)^2\,\sigma_2^2 + 2w\,(1 - w)\,\sigma_{12}$$

Understanding the Diversification Gains

- The optimal w can be solved by minimizing this expression, and is

$$w^\star = \frac{\sigma_2^2 - \sigma_{12}}{\sigma_1^2 + \sigma_2^2 - 2\sigma_{12}}, \; 1 - w^\star = \frac{\sigma_1^2 - \sigma_{12}}{\sigma_1^2 + \sigma_2^2 - 2\sigma_{12}}$$

- Intuition is that the weight on a model is higher the:
  - Larger the variance of the other model
  - Lower the correlation between the models
- 1 weight will be larger than 1 if $\rho \geq \frac{\sigma_2}{\sigma_1}$
- Weights will be equal if $\sigma_1 = \sigma_2$ for any value of correlation
  - Intuitively this must be the case since model 1 and 2 are indistinguishable from a MSE point-of-view
  - When will "optimal" combinations out-perform equally weighted combinations? Any time $\sigma_1 \neq \sigma_2$
- If $\rho = 1$ then only select model with lowest variance (mathematical formulation is not well posed in this case)

Constrained weights

- The previous optimal weight derivation did not impose any restrictions on the weights
- In general some of the weights will be negative, and some will exceed 1
- Many combinations are implemented in a relative, constrained scheme

$$\min_{\mathbf{w}} \mathrm{E}\left[e^2\right] = \mathrm{E}\left[\left(y_{t+h} - \mathbf{w}'\hat{\mathbf{y}}\right)^2\right] \text{ subject to } \mathbf{w}'\boldsymbol{\iota} = 1$$

- The intercept is omitted (although this isn't strictly necessary)
- If the biases are all 0, then the solution is dual to the usual portfolio minimization problem, and is given by

$$\mathbf{w}^\star = \frac{\boldsymbol{\Sigma}_{\hat{\mathbf{y}}\hat{\mathbf{y}}}^{-1}\boldsymbol{\iota}}{\boldsymbol{\iota}'\boldsymbol{\Sigma}_{\hat{\mathbf{y}}\hat{\mathbf{y}}}^{-1}\boldsymbol{\iota}}$$

- This solution is the same as the Global Minimum Variance Portfolio

Combinations as Hedge against Structural Breaks

UNIVERSITY OF OXFORD

- One often cited advantage of combinations is (partial) robustness to structural breaks
- Best case is if two positively correlated variables have shifts in opposite directions
- Combinations have been found to be more stable than individual forecasts
  - This is mostly true for static combinations
  - Dynamic combinations can be unstable since some models may produce large errors from time-to-time

Weight Estimation

- All discussion has focused on "optimal" weights, which requires information on the mean and covariance of both y_{t+h} and $\hat{\mathbf{y}}_{t+h|t}$
  - This is clearly highly unrealistic
- In practice weights must be estimated, which introduces extra estimation error
- Theoretically, there should be no need to combine models when all forecasting models are generated by the econometrician (e.g. when using $\mathcal{F}^C$)
- In practice, this does not appear to be the case
  - High dimensional search space for "true" model
  - Structural instability
  - Parameter estimation error
  - Correlation among predictors

Clemen (1989): "Using a combination of forecasts amounts to an admission that the forecaster is unable to build a properly specified model"

Weight Estimation

- Whether a combination is needed is closely related to forecast encompassing tests
- Model averaging can be thought of a method to avoid the risk of model selection
  - Usually important to consider models with a wide range of features and many different model selection methods
- Has been consistently documented that *prescreening* models to remove the worst performing is important before combining
- One method is to use the SIC to remove the worst models
  - Rank models by SIC, and then keep the x% best
- Estimated weights are usually computed in a 3rd step in the usual procedure
  - R: Regression
  - P: Prediction
  - S: Combination estimation
  - $T = P + R + S$
- Many schemes have been examined

Weight Estimation

- Standard least squares with an intercept

$$y_{t+h} = w_0 + \mathbf{w}'\hat{\mathbf{y}}_{t+h|t} + \epsilon_{t+h}$$

- Least squares without an intercept

$$y_{t+h} = \mathbf{w}'\hat{\mathbf{y}}_{t+h|t} + \epsilon_{t+h}$$

- Linearly constrained least squares

$$y_{t+h} - \hat{y}_{t+h,n|t} = \sum_{i=1}^{n-1} w_i \left(\hat{y}_{t+h,i|t} - \hat{y}_{t+h,n|t}\right) + \epsilon_{t+h}$$

  - This is just a constrained regression where $\sum w_i = 1$ has been implemented where $w_n = 1 - \sum_{i=1}^{n-1} w_i$
  - Imposing this constraint is thought to help when the forecast is persistent

$$e^c_{t+h|t} = -w_0 + \left(1 - \mathbf{w}'\boldsymbol{\iota}\right) y_{t+h} + \mathbf{w}'\mathbf{e}_{t+h|t}$$

  - $\mathbf{e}_{t+h|t}$ are the forecasting errors from the n models
  - Only matters if the forecasts may be biased

Weight Estimation

- Constrained least squares

$$y_{t+h} = \mathbf{w}'\hat{\mathbf{y}}_{t+h|t} + \epsilon_{t+h} \text{ subject to } \mathbf{w}'\boldsymbol{\iota}=1,\ w_i \geq 0$$

  - This is not a standard regression, but can be easily solved using quadratic programming (MATLAB `quadprog`)

- Forecast combination where the covariance of the forecast errors is assumed to be diagonal
  - Produces weights which are all between 0 and 1
  - Weight on forecast i is

$$w_i = \frac{\frac{1}{\sigma_i^2}}{\sum_{j=1}^{n} \frac{1}{\sigma_j^2}}$$

  - May be far from optimal if ρ is large
  - Protects against estimator error in the covariance

Weight Estimation

- Median
  - Can use the median rather than the mean to aggregate
  - Robust to outliers
  - Still suffers from not having any reduction in parameter variance in the actual forecast
- Rank based schemes
  - Weights are inversely proportional to model's rank

$$w_i = \frac{\mathcal{R}_{t+h,i|t}^{-1}}{\sum_{j=1}^{n} \mathcal{R}_{t+h,j|t}^{-1}}$$

  - Highest weight to best model, ratio of weights depends only on relative ranks
  - Places relatively high weight on top model
- Probability of being the best model-based weights
  - Count the proportion that model i outperforms the other models

$$p_{t+h,i|t} = T^{-1} \sum_{t=1}^{T} \cap_{j=1,j\neq i}^{n} I\left[L\left(e_{t+h,i|t}\right) < L\left(e_{t+h,j|t}\right)\right]$$

$$y_{t+h|t}^{c} = \sum_{i=1}^{n} p_{t+h,i|t} \hat{y}_{t+h,i|t}$$

Broad Recommendations

- Simple combinations are difficult to beat
  - $1/n$ often outperforms estimated weights
  - Constant usually beat dynamic
  - Constrained outperform unconstrained (when using estimated weights)
- Not combining and using the best fitting performs worse than combinations – often substantially
- Trimming bad models prior to combining improves results
- Clustering similar models (those with the highest correlation of their errors) *prior* to combining leads to better performance, especially when estimating weights
  - Intuition: Equally weighted portfolio of models with high correlation, weight estimation using a much smaller set with lower correlations
- Shrinkage improves weights when estimated
- If using dynamic weights, shrink towards static weights

Equal Weighting

UNIVERSITY OF OXFORD

- Equal weighting is hard to beat when the variance of the forecast errors are similar
- If the variance are highly heterogeneous, varying the weights is important
  - If for nothing else than to down-weight the forecasts with large error variances
- Equally weighted combinations are thought to work well when models are unstable
  - Instability makes finding "optimal" weights very challenging
- Trimmed equally-weighted combinations appear to perform better than equally weighted, at least if there are some very poor models
  - May be important to trim both "good" and "bad" models (in-sample performance)
    - Good models are over-fit
    - Bad models are badly mis-specified

Shrinkage Methods

- Linear combination

$$\hat{y}^c_{t+h|t} = \mathbf{w}'\hat{\mathbf{y}}_{t+h|t}$$

  Standard least squares estimates of combination weights are very noisy
- Often found that "shrinking" the weights toward a *prior* improves performance
- Standard prior is that $w_i = \frac{1}{n}$
- However, do not want to be *dogmatic* and so use a distribution for the weights
- Generally for an arbitrary *prior weight* $\mathbf{w}_0$,

$$\mathbf{w}|\tau^2 \sim N(\mathbf{w}_0, \mathbf{\Omega})$$

- $\mathbf{\Omega}$ is a correlation matrix and τ^2 is a parameter which controls the amount of shrinkage

Shrinkage Methods

- Leads to a weighted average of the prior and data

$$\bar{\mathbf{w}} = \left(\boldsymbol{\Omega} + \hat{\mathbf{y}}'\hat{\mathbf{y}}\right)^{-1}\left(\boldsymbol{\Omega}\mathbf{w}_0 + \hat{\mathbf{y}}'\hat{\mathbf{y}}\hat{\mathbf{w}}\right)$$

- $\hat{\mathbf{w}}$ is the usual least squares estimator of the optimal combination weight
- If $\boldsymbol{\Omega}$ is very large compared to $\mathbf{y}'\mathbf{y} = \sum_{t=1}^{T} \mathbf{y}_{t+h|t}\mathbf{y}'_{t+h|t}$ then $\bar{\mathbf{w}} \approx \mathbf{w}_0$
- On the other hand, if $\mathbf{y}'\mathbf{y}$ dominates, then $\bar{\mathbf{w}} \approx \hat{\mathbf{w}}$
- Other implementation use a g-prior, which is scalar

$$\bar{\mathbf{w}} = \left(g\hat{\mathbf{y}}'\hat{\mathbf{y}} + \hat{\mathbf{y}}'\hat{\mathbf{y}}\right)^{-1}\left(g\hat{\mathbf{y}}'\hat{\mathbf{y}}\mathbf{w}_0 + \hat{\mathbf{y}}'\hat{\mathbf{y}}\hat{\mathbf{w}}\right)$$

- Large values of $g \geq 0$ least to large amounts of shrinkage
- 0 corresponds to OLS

$$\bar{\mathbf{w}} = \mathbf{w}_0 + \frac{\hat{\mathbf{w}} - \mathbf{w}_0}{1+g}$$

Inference for Many Forecasts

- Six papers:
  - White, H. “A reality check for data snooping”. *Econometrica*
  - Hansen, P. “A Test for Superior Predictive Ability”. *JBES*
  - Sullivan, Timmermann & White. “Data-Snooping, Technical Trading Rule Performance, and the Bootstrap”. *Journal of Finance*
  - Romano & Wolf. “Stepwise Multiple Testing as Formalized Data Snooping”. *Econometrica*
  - Hansen, Lunde & Nason. “The Model Confidence Set”. *Econometrica*
  - Bajgrowicz & Scaillet. “Technical trading revisited: false discoveries, persistence tests and transaction costs”. *Journal of Financial Economics*

Diebold-Mariano-West

- The Diebold-Mariano-West test examines whether two forecasts have equal predictive ability
- DMW tests are all based on the difference of two loss functions

$$\delta_t = L\left(y_{t+h}, \hat{y}^A_{t+h|t}\right) - L\left(y_{t+h}, \hat{y}^B_{t+h|t}\right)$$

- The test statistic is based on the asymptotic normality of $\bar{\delta} = P^{-1}\sum_{t=R+1}^{T}\delta_t$
- If $P/R \to 0$ then

$$\sqrt{P}\left(\bar{\delta} - \mathrm{E}\left[\delta\right]\right) \xrightarrow{d} N\left(0, \sigma^2\right)$$

- σ^2 is the long-run variance, that is

$$\sigma^2 = \lim_{P\to\infty} \mathrm{V}\left[P^{-\frac{1}{2}}\sum_{t=R+1}^{T}\delta_t\right]$$

- Must account for autocovariances, so a HAC estimator is used (Newey-West)

DMW with the Bootstrap

- Alternatively could estimate the variance using the bootstrap
- For example, the stationary bootstrap could be used as long as the window length grows with the size of the evaluation sample
- To implement the stationary bootstrap, the loss differentials would be directly re-sampled to construct $\bar{\delta}_b^*$ for $b = 1, \ldots, B$
- The variance would then be computed as

$$\hat{\sigma}_{BS}^2 = \frac{P}{B} \sum_{b=1}^{b} \left(\bar{\delta}_b^\star - \bar{\delta} \right)^2$$

- The test statistic is then

$$DMW = \frac{\bar{\delta}}{\sqrt{\hat{\sigma}_{BS}^2}}$$

  - Note: the $\sqrt{P}$ term is implicit in the denominator since σ_{BS}^2 will decline as the sample size grows ($\hat{\sigma}_{BS}^2 \approx \hat{\sigma}^2/P$)

DMW using percentile method

- Alternatively, inference could be made using the percentile method
- To implement the percentile method, it is necessary to enforce the null $H_0 : \mathrm{E}[\delta_t] = 0$
- This can be done by re-centering the loss differentials around the average in the data: $\tilde{\delta}_t = \delta_t - \bar{\delta}$
- The centered loss differentials $\tilde{\delta}_t$ could then be re-sampled to compute an estimate of the average loss-differential $\bar{\tilde{\delta}}^{\star}_b$
- Inference using the percentile method would be based on the empirical frequency where $\bar{\delta} < \bar{\tilde{\delta}}^{\star}_b$ or $\bar{\delta} > \bar{\tilde{\delta}}^{\star}_b$

DMW using percentile method

- Since the test is 2-sided||

$$2 \times B^{-1} \sum_{b=1}^{B} I\left[\left|\bar{\bar{\delta}}_b^{\star}\right| < \left|\bar{\delta}\right|\right]$$

  - If many of the re-sampled centered means are less then $\bar{\delta}$, then the loss differential does not appear large
  - If few of the re-sampled centered means are less than $\bar{\delta}$, then the loss differential appears large

- Since the distribution is asymptotically normal, there is no need to use the percentile method since the bootstrap t-stat is simple to construct

Reality Check

- The *Reality Check* extends DMW to testing for *Superior Predictive Ability* (SPA)
- Tests of SPA examine whether a set of forecasting models can outperform a benchmark
- Suppose forecasts were available for m forecasts, $j = 1, \ldots, m$
- The vector of loss differentials *relative to a benchmark* could be constructed as

$$\boldsymbol{\delta}_t = \begin{bmatrix} L\left(y_{t+h}, \hat{y}_{t+h,BM|t}\right) - L\left(y_{t+h}, \hat{y}_{t+h,1|t}\right) \\ L\left(y_{t+h}, \hat{y}_{t+h,BM|t}\right) - L\left(y_{t+h}, \hat{y}_{t+h,2|t}\right) \\ \vdots \\ L\left(y_{t+h}, \hat{y}_{t+h,BM|t}\right) - L\left(y_{t+h}, \hat{y}_{t+h,m|t}\right) \end{bmatrix}$$

- $\hat{y}_{t+h,BM|t}$ is the loss from the *benchmark forecast*

Asymptotic distribution in the RC

- Under similar arguments as in Diebold & Mariano and West,

$$\sqrt{P}\left(\bar{\boldsymbol{\delta}} - \mathrm{E}\left[\bar{\boldsymbol{\delta}}\right]\right) \xrightarrow{d} N\left(\mathbf{0}, \boldsymbol{\Sigma}\right)$$

- $\boldsymbol{\Sigma}$ is the asymptotic covariance matrix of the average loss differentials

$$\boldsymbol{\Sigma} = \lim_{P \to \infty} \mathrm{V}\left[P^{-\frac{1}{2}} \sum_{t=R+1}^{T} \boldsymbol{\delta}_t\right]$$

- This looks virtually identical to the case of the univariate DMW test

Hypotheses of SPA

- If the benchmark model is as good as the other models, then the mean of each element of $\boldsymbol{\delta}_t$ should be 0 *or* negative
  - These are *losses*, so if the BM is better, then its loss is smaller then the loss from the other model
- A total of m models
- The null in a test of SPA is

$$H_0 : \max_{j=1,\ldots,m} \left(\mathrm{E} \left[\delta_{j,t} \right] \right) \leq 0$$

- The alternative is the natural one,

$$H_1 : \max_{j=1,\ldots,m} \left(\mathrm{E} \left[\delta_{j,t} \right] \right) > 0$$

- ***Note**: If no models are statistically better than the benchmark, then there is no point in implementing the RC*

Examples of SPA: MSE

- The standard example is for comparing models using MSE (or MAE, or similar)

$$L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right) = \left(y_{t+h} - \hat{y}_{t+h,j|t}\right)^2$$

- The vector of loss differentials is then

$$\boldsymbol{\delta}_t = \begin{bmatrix} \left(y_{t+h} - \hat{y}_{t+h,BM|t}\right)^2 - \left(y_{t+h} - \hat{y}_{t+h,1|t}\right)^2 \\ \left(y_{t+h} - \hat{y}_{t+h,BM|t}\right)^2 - \left(y_{t+h} - \hat{y}_{t+h,2|t}\right)^2 \\ \vdots \\ \left(y_{t+h} - \hat{y}_{t+h,BM|t}\right)^2 - \left(y_{t+h} - \hat{y}_{t+h,m|t}\right)^2 \end{bmatrix}$$

- This is the simplest form of an SPA test

Examples of SPA: Return Predictability

- SPA can also be used to test whether the returns of a set of trading models are equal
- In this case the "loss" function is the *negative* of the return from the strategy

$$L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right) = -\ln\left(1 + y_{t+h} S\left(\hat{y}_{t+h,j|t}\right)\right)$$

- $S\left(\hat{y}_{t+h,j|t}\right)$ is a signal which indicates the size of the portfolio
  - y_{t+h} is the holding period return of the asset
  - Could be -1, 0, 1 for short, out, long strategies
  - $\hat{y}_{t+h,j|t}$ is the input for the signal function, e.g. a Moving Average Oscillator
- The vector of loss differentials is then

$$\boldsymbol{\delta}_t = \begin{bmatrix} \ln\left(1 + y_{t+h} S\left(\hat{y}_{t+h,1|t}\right)\right) - \ln\left(1 + y_{t+h} S\left(\hat{y}_{t+h,BM|t}\right)\right) \\ \vdots \\ \ln\left(1 + y_{t+h} S\left(\hat{y}_{t+h,m|t}\right)\right) - \ln\left(1 + y_{t+h} S\left(\hat{y}_{t+h,BM|t}\right)\right) \end{bmatrix}$$

- The benchmark could be a simple strategy, e.g. buy-and-hold ($S(\cdot) = 1$)
- Ultimately the "loss differential" is the difference between the returns of a set of strategies and the benchmark strategy

Example: Predictive Likelihood

- SPA can be used to test distribution fit
- The loss function is just the *negative* of the likelihood

$$L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right) = -l_j\left(y_{t+h}|\hat{y}_{t+h,j|t}\right)$$

  - $\hat{y}_{t+h,j|t}$ contains any time-t information needed to compute the log-likelihood
- The vector of loss differentials is then

$$\boldsymbol{\delta}_t = \begin{bmatrix} l_1\left(y_{t+h}|\hat{y}_{t+h,1|t}\right) - l_{BM}\left(y_{t+h}|\hat{y}_{t+h,BM|t}\right) \\ l_2\left(y_{t+h}|\hat{y}_{t+h,2|t}\right) - l_{BM}\left(y_{t+h}|\hat{y}_{t+h,BM|t}\right) \\ \vdots \\ l_m\left(y_{t+h}|\hat{y}_{t+h,m|t}\right) - l_{BM}\left(y_{t+h}|\hat{y}_{t+h,BM|t}\right) \end{bmatrix}$$

- The benchmark could be a simple strategy, e.g. buy-and-hold ($S(\cdot) = 1$)
- Ultimately the differential is just the difference between the returns of a set of strategies and the benchmark strategy

Example: α from a multifactor model

- Suppose you were interested in testing for excess performance
- Usual APT type regression

$$r^e_{j,t} = \alpha_j + \mathbf{f}'_t \boldsymbol{\beta}_j + \epsilon_{j,t}$$

- The "benchmark α" is 0 – the test is implemented directly on the estimated αs
- Loss function is just $-\hat{\alpha}$ (*negative* excess performance)
- The vector of loss differentials is then

$$\boldsymbol{\delta}_t = \begin{bmatrix} r^e_{1,t} - \mathbf{f}'_t \hat{\boldsymbol{\beta}}_1 \\ \vdots \\ r^e_{m,t} - \mathbf{f}'_t \hat{\boldsymbol{\beta}}_m \end{bmatrix} = \begin{bmatrix} \hat{\alpha}_1 + \hat{\epsilon}_{1,t} \\ \vdots \\ \hat{\alpha}_m + \hat{\epsilon}_{m,t} \end{bmatrix}$$

- Used to test fund manager skill

Implementing the Reality Check

- The Reality Check is implemented using the P by m matrix of loss differentials
  - P out-of-sample periods
  - m models
- The original article describes two methods
  - Monte Carlo Reality Check
  - Bootstrap Reality Check
- In practice, only the Bootstrap Reality Check is used
- The distribution of the *maximum* of normals is not normal, and so only the percentile method is applicable

UNIVERSITY OF OXFORD

Algorithm (Bootstrap Reality Check)

1. *Compute* $T^{RC} = \max\left(\bar{\delta}\right)$
2. *For* $b = 1, \ldots, B$ *re-sample the vector of loss differentials* $\boldsymbol{\delta}_t$ *to construct a bootstrap sample* $\left\{\boldsymbol{\delta}^{\star}_{b,t}\right\}$ *using the stationary bootstrap*
3. *Using the bootstrap sample, compute*

$$T^{\star RC}_b = \max\left(P^{-1}\sum_{t=R+1}^{T}\boldsymbol{\delta}^{\star}_{b,t} - \bar{\boldsymbol{\delta}}\right)$$

4. *Compute the Reality Check p-value as the percentage of the bootstrapped maxima which are larger than the sample maximum*

$$p - value = B^{-1}\sum_{b=1}^{b} I\left[T^{\star RC}_b > T^{RC}\right]$$

Intuition

- The bootstrap means are like draws (simulation) from the asymptotic distribution $N(\mathbf{0}, \mathbf{\Sigma})$
- Taking the maximum of these draws simulates the distribution of a set of correlated normals
- Each bootstrap mean is centered at the sample mean
  - This is known as using the *Least Favorable Configuration* (LFC) point
  - Simulation is done assuming any model could as good as the benchmark
- Since the asymptotic distribution can be simulated, asymptotic critical values and p-values can be constructed directly
- The Monte Carlo Reality Check works by first estimating $\mathbf{\Sigma}$ using a HAC estimator, and then simulating random normals directly
  - MCRC is equivalent to BRC, only requires estimating:
    - A potentially large covariance is m is big
    - The Choleski decomposition of this covariance
    - B drawn from this Choleski
  - In practice, m may be so large that the covariance matrix won't fit in a normal computer's memory

Revisiting: α from a multifactor model

- The original formulation had

$$\boldsymbol{\delta}_t = \begin{bmatrix} r^e_{1,t} - \mathbf{f}'_t \hat{\boldsymbol{\beta}}_1 \\ \vdots \\ r^e_{m,t} - \mathbf{f}'_t \hat{\boldsymbol{\beta}}_m \end{bmatrix} = \begin{bmatrix} \hat{\alpha}_1 + \hat{\epsilon}_{1,t} \\ \vdots \\ \hat{\alpha}_m + \hat{\epsilon}_{m,t} \end{bmatrix}$$

- Alternatively distribution could be built up by directly re-sampling the returns and factors jointly
- This would allow $T_b^{\star RC} = \max_{j=1,\ldots,m} \left(\alpha^*_{j,b} - \hat{\alpha}_j \right)$ to be computed from a cross-sectional regression in each bootstrap
- Reality check allow for parameter estimation error as long as $(P/R) \ln \ln R \to 0$ which is similar to $P/R \to 0$
- Also works if $P/R \to \infty$, in which case it is essential to re-sample returns and factors and re-estimate $\hat{\boldsymbol{\beta}}^{\star}_{j,b}$ in each bootstrap

Application in Original Paper

- The original paper is applied to the BLL-type trading rules
- Used S&P 500 rather than DJIA
- Constructed 4 types of trading rule primitives:
  - Momentum measures: $\left(p_t - p_{t-j}\right)/p_{t-j}$ for $j \in \{1, \ldots, 11\}$ (11 rules)
  - Trend: $p_{t-i} = \alpha + \beta\left(m - i\right) + \epsilon_j$ for $m \in \{5, 10, 15, 20\}$ day periods (4 rules)
  - Relative strength: $\tau^{-1} \sum_{i=-\tau+1}^{0} I\left[\left(p_{t-i} - p_{t-i-1}\right) > 0\right]$ for $\tau \in \{5, 10, 15, 20\}$(4 rules)
  - Moving average oscillator for fast speeds of $\{1, 5, 10, 15\}$ and slow speeds of $\{5, 10, 15, 20\}$ (10 rules)
    - Note: Slow has to be strictly longer than fast, so a total of $4 + 3 + 2 + 1 = 10$ rules
- All combinations of 3 of these 29 variables were fed into a linear regression to produce forecasts

$$r_{t+1} = \beta_1 + \beta_2 x_{i,t} + \beta_3 x_{j,t} + \beta_4 x_{k,t} + \epsilon_{t+1}$$

- For $i, j, k \in \{1, \ldots, 29\}$ without repetition, so ${}_{29}C_3 = 3654$ rules

Application in Original Paper

- Benchmark is a model which includes only a constant

$$r_{t+1} = \beta_1 + \epsilon_{t+1}$$

- Models compared in terms of MSE

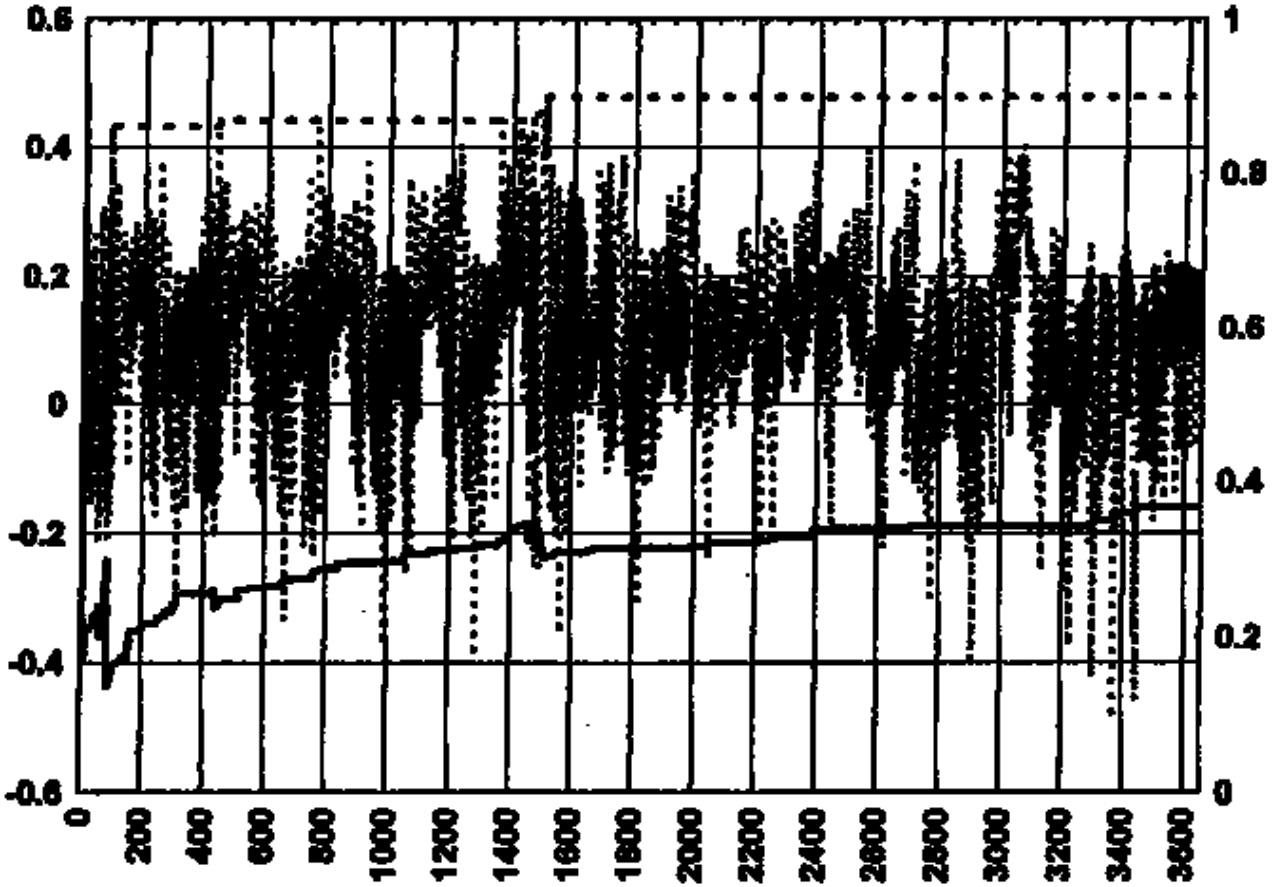
$$L\left(y_{t+1}, \hat{y}_{t+1|t}\right) = \left(y_{t+1} - \hat{\beta}_0 - \hat{\beta}_1 x_{i,t} - \hat{\beta}_2 x_{j,t} - \hat{\beta}_3 x_{k,t}\right)^2$$

- Models also compared in terms of directional accuracy

$$L\left(y_{t+1}, \hat{y}_{t+1|t}\right) = -I\left[y_{t+1}\left(\hat{\beta}_0 + \hat{\beta}_1 x_{i,t} + \hat{\beta}_2 x_{j,t} + \hat{\beta}_3 x_{k,t}\right) > 0\right]$$

  - The negative is used to turn a "good" (same sign) into a "bad"
  - Modification allows application of RC without modification since null is $H_0 : \max\left(\mathrm{E}\left[\delta_{j,t}\right]\right) \leq 0$

MSE Differential



$\max \bar{\delta}_i$

RC P-val

Experiment Number

- Negative MSE differential plotted (higher is better)

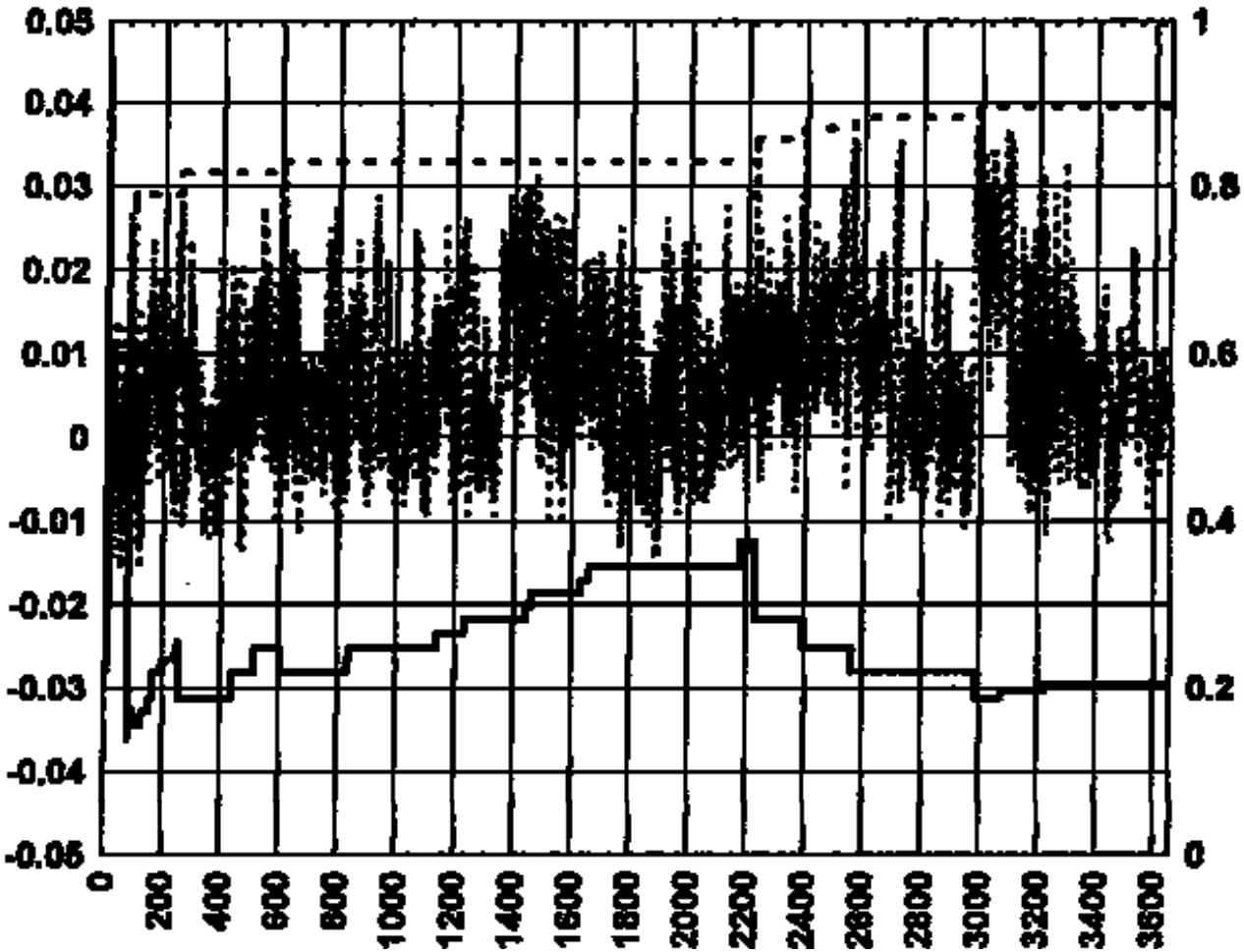
Sign Prediction

REALITY CHECK RESULTS: DIRECTIONAL ACCURACY PERFORMANCE

Best predictor variables: $Z_{t,13}$, $Z_{t,14}$, $Z_{t,26}$

	Best Experiment	Benchmark
Percent Correct	54.7493	50.7916
Difference in Prediction Directional Accuracy:	.0396	
Bootstrap Reality Check p-value:	.2040	
Naive p-value:	.0036	

Sign Prediction



$\max \bar{\delta}_i$

RC P-val

Experiment Number

Hansen's Test of SPA

- Hansen (2005, *JBES*) provided two refinements of the RC
  1. Studentized loss differentials
  2. Omission of very bad models from the distribution of the test statistic
- From a practical point-of-view, the first is a very important consideration
- From a theoretical point-of-view, the second is the important issue
  - The second can be ignored if no models are very poor
  - This may be difficult if using automated model generation schemes

Studentization of Loss Differentials

- The RC uses the loss differentials directly
- This can lead to a loss of power if there is a large amount of cross-sectional heteroskedasticity
- Bad, high variance model can mask a good, low variance model
- The solution is to use the Studentized loss differential
- The test statistic is is based on

$$T^{SPA} = \max_{j=1,\ldots,m} \left(\frac{\bar{\delta}_j}{\sqrt{\hat{\omega}_j^2 / P}} \right)$$

- $\hat{\omega}_j^2$ is an estimator of the asymptotic (long-run) variance of $\bar{\delta}_j$

$$\hat{\omega}_j^2 = \hat{\gamma}_{j,0} + 2 \sum_{i=1}^{P-1} k_i \hat{\gamma}_{j,i}$$

  - $\gamma_{j,i}$ is the i$^{\text{th}}$ sample autocovariance of the sequence $\{\delta_{j,t}\}$
  - $k_i = \frac{P-i}{P} \left(1 - \frac{1}{w}\right)^i + \frac{i}{P} \left(1 - \frac{1}{w}\right)^{P-i}$ where w is the window length in Stationary Bootstrap)

- Alternatively use bootstrap variance $\hat{\omega}_j^2 = \frac{P}{B} \sum_{b=1}^{B} \left(\bar{\delta}_{b,j}^{\star} - \bar{\delta}_j \right)^2$

Algorithm (Studentized Bootstrap Reality Check)

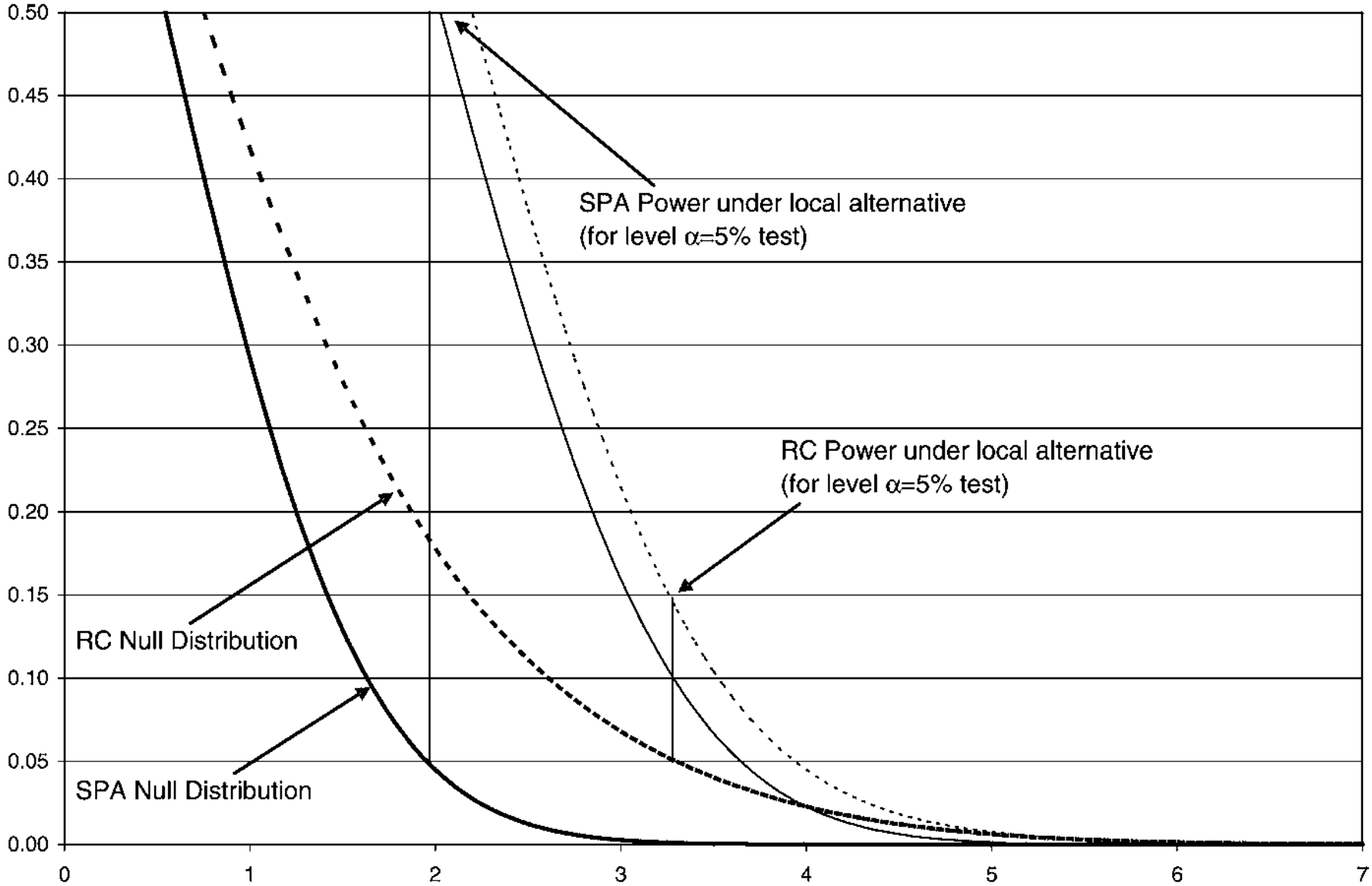
1. *Estimate* $\hat{\omega}_j^2$ *and compute* $T_u^{SPA} = \max\left(\bar{\delta}/\sqrt{\hat{\omega}_j^2/P}\right)$
2. *For* $b = 1, \ldots, B$ *re-sample the vector of loss differentials* $\boldsymbol{\delta}_t$ *to construct a bootstrap sample* $\left\{\boldsymbol{\delta}_{b,t}^{\star}\right\}$ *using the stationary bootstrap*
3. *Using the bootstrap sample, compute*

$$T_{u,b}^{\star SPA} = \max\left(\frac{P^{-1}\sum_{t=R+1}^{T}\delta_{j,b,t}^{\star} - \bar{\delta}_j}{\sqrt{\hat{\omega}_j^2/P}}\right)$$

4. *Compute the Studentized Reality Check p-value as the percentage of the bootstrapped maxima which are larger than the sample maximum*

$$p - value = B^{-1}\sum_{b=1}^{b} I\left[T_{u,b}^{\star SPA} > T_u^{SPA}\right]$$

Gains from Studentization



inus) The cdfs for the Test Statistics T^{RC} and T^{SPA} Under the Null Hypothesis $\mu_1 = \mu_2 = 0$ and the

The u in T_u^{SPA} is for *upper*

- The U is included to indicate that the p-value derived using the LFC may not be the best p-value
- Suppose the some of the models have a very low mean and a high standard deviation
- In the RC and SPA-U, all models are assumed to be as good as the benchmark
- This is implemented by always re-centering the bootstrap samples around $\bar{\delta}_j$
- If a model is rejectably bad, then it may be possible to improve the power of the RC/SPA-U by excluding this model
- This is implemented using a "pre-test" of the form

$$I_j^u = 1, \quad I_j^c = \frac{\bar{\delta}_j}{\sqrt{\hat{\omega}_j^2 / P}} > -\sqrt{2 \ln \ln P}, \quad I_j^l = \bar{\delta}_j > 0$$

  - The first (c for *consistent*) tests whether the standardized mean loss differential is greater than a HQ-like lower bound
  - The second (l for *lower*) only re-centers if the loss-differential is positive (e.g. the benchmark is out-performed)

Algorithm (Test of SPA)

1. *Estimate* $\hat{\omega}_j^2$ *and compute* $T^{SPA} = \max\left(\bar{\delta}/\sqrt{\hat{\omega}_j^2/P}\right)$
2. *For* $b = 1, \ldots, B$ *re-sample the vector of loss differentials* $\boldsymbol{\delta}_t$ *to construct a bootstrap sample* $\left\{\boldsymbol{\delta}_{b,t}^{\star}\right\}$ *using the stationary bootstrap*
3. *Using the bootstrap sample, compute*

$$T_{s,b}^{\star SPA} = \max\left(\frac{P^{-1}\sum_{t=R+1}^{T}\delta_{j,b,t}^{\star} - I_j^s\bar{\delta}_j}{\sqrt{\hat{\omega}_j^2/P}}\right), \quad s = l, c, u$$

4. *Compute the Studentized Reality Check p-value as the percentage of the bootstrapped maxima which are larger than the sample maximum*

$$p - value = B^{-1}\sum_{b=1}^{b} I\left[T_{s,b}^{\star SPA} > T^{SPA}\right], \quad s = l, u, c$$

Comments on SPA

- The three versions only differ on whether a model is re-centered
- If a model is *not* re-centered, then it is unlikely to be the maximum in the re-sample distribution
  - This is how "bad" models are discarded in the SPA
- Can compute 6 different p-values statistics
  - Studentized or unmodified
  - Indicator function in l, c, u
    - Test statistic does not depend on l, c, u, only p-value does
- Reality Check uses unmodified loss differentials and u
- In practice Studentization beings important gains
- Using c is important if using SPA on large universe of automated rules if some may be very poor

Power Gains in SPA from Re-centering

These are easily obtained using the "estimators," $\hat{\boldsymbol{\mu}}^l$ and $\hat{\boldsymbol{\mu}}^u$ given by $\hat{\mu}_k^l \equiv \min(\bar{d}_k, 0)$ and $\hat{\mu}_k^u = 0$, $k = 1, \ldots, m$, where the latter yields the LFC-based test. It is simple to verify that $\hat{\boldsymbol{\mu}}^l \le \hat{\boldsymbol{\mu}}^c \le \hat{\boldsymbol{\mu}}^u$, which in part motivates the superscripts, and we have the following result, where F_0 is the cdf of $\varphi(\mathbf{Z}, \mathbf{v}_0)$ that we defined in Theorem 1.

Theorem 2. Let F_n^i be the cdf of $\varphi(n^{1/2}\mathbf{Z}_n^i, \mathbf{V}_n)$, for $i = l, c,$ or u, where $n^{1/2}(\mathbf{Z}_n^i - \hat{\boldsymbol{\mu}}^i) \xrightarrow{d} N_m(\mathbf{0}, \boldsymbol{\Omega})$. Suppose that Assumptions 1 and 2 hold; then $F_n^c \to F_0$ as $n \to \infty$, for all continuity points of F_0 and $F_n^l(x) \le F_n^c(x) \le F_n^u(x)$ for all n and all $x \in \mathbb{R}$.

Theorem 2 demonstrates that $\hat{\boldsymbol{\mu}}^c$ leads to a consistent estimate of the asymptotic distribution of our test statistic. The theorem also demonstrates that $\hat{\boldsymbol{\mu}}^l$ and $\hat{\boldsymbol{\mu}}^u$ provide upper and lower bound for the distribution F_n^c that can be useful in practice; for example, a substantial difference between these bounds is indicative of the presence of poor alternatives, in which case the sample-dependent null distribution is useful.

Given a value for the test statistic $t = T_n(\mathbf{d}_1, \ldots, \mathbf{d}_n)$, it is natural to define the *true asymptotic p value* as $p_0(t) \equiv 1 - F_0(t)$. The empirical p value is deduced from an estimate of F_n^i, $i = l, c, u$, and the following corollary demonstrates that $\hat{\boldsymbol{\mu}}^c$ yields a consistent p value.

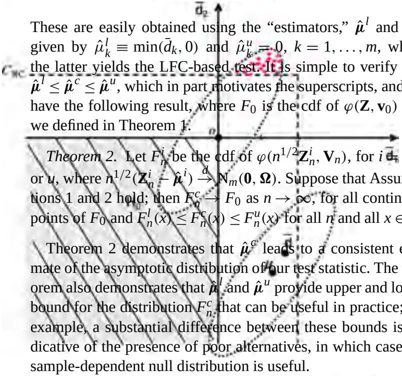


Figure 2. A Situation Where the RC Fails to Reject a False Null Hypothesis. The true parameter value is $\boldsymbol{\mu} = (\mu_1, \mu_2)'$, the sample estimate is $\bar{\mathbf{d}} = (\bar{d}_1, \bar{d}_2)'$, and C_{RC} represents the critical value derived from a null distribution that tacitly assumes that $\boldsymbol{\mu} = (0, 0)'$.

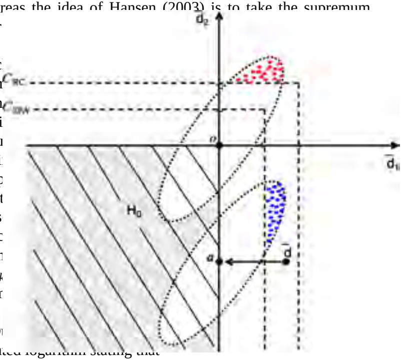


Figure 3. How the Power Is Improved by Using the Sample-Dependent Null Distribution. This distribution is centered about $\hat{\boldsymbol{\mu}}^c = a$, which leads to the critical value C_{SPA}. In contrast, the RC fails to reject the null hypothesis, because the LFC-based null distribution leads to the larger critical value C_{RC}.

whereas the idea of Hansen (2003) is to take the supremum over ...

$$P\left(\limsup_{n\to\infty} \frac{n^{1/2}(\bar{d}_k - \mu_k)}{\omega_k} = +\sqrt{2\log\log n}\right) = 1.$$

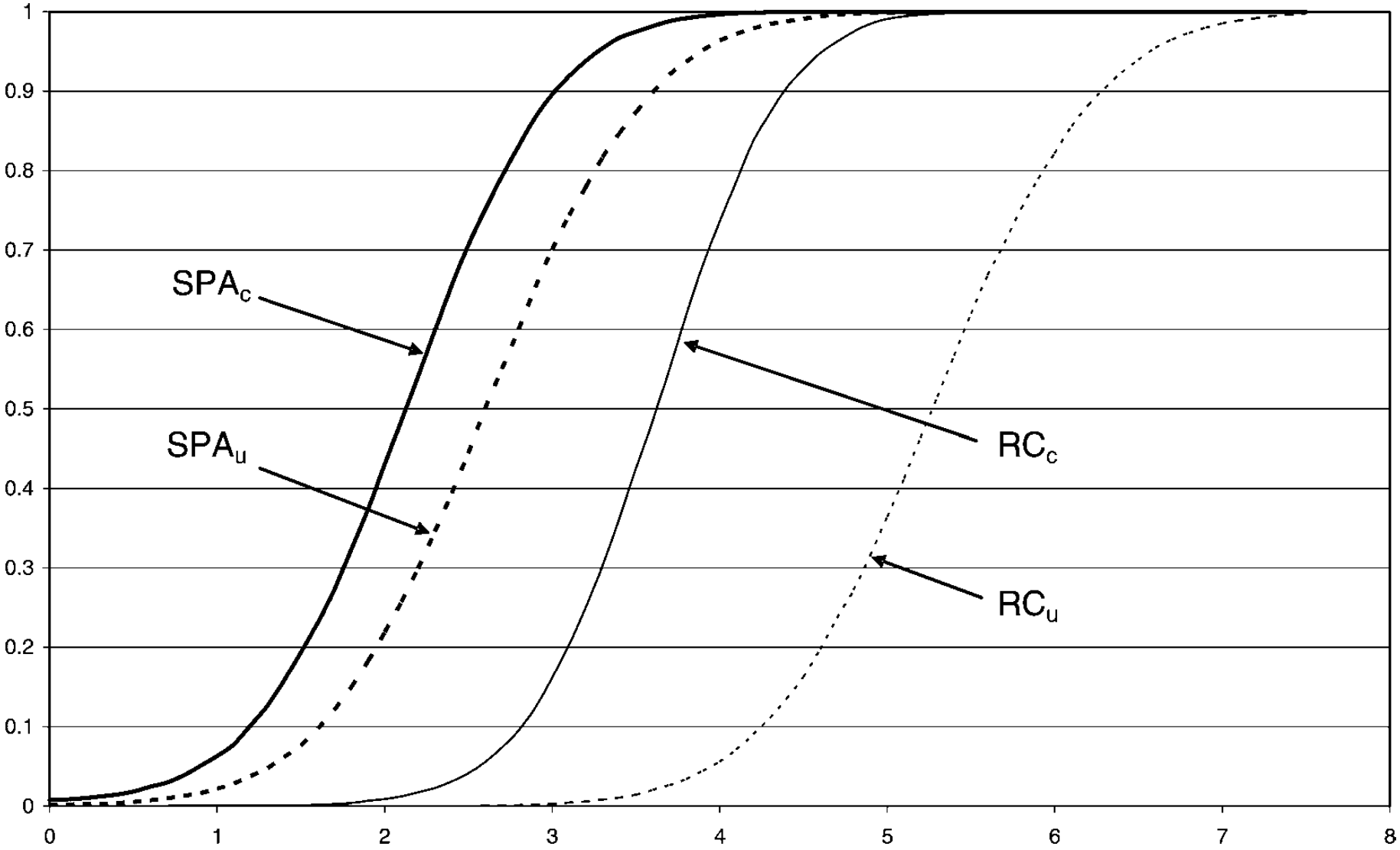
alternatives in the analysis. Naturally, we would want to avoid such properties to the extent possible.

The first equality shows that $\hat{\mu}_k^c$ effectively captures all of the elements of $\boldsymbol{\mu}$ that are 0, such that $\mu_k = 0 \Rightarrow \hat{\mu}_k^c = 0$ almost surely. Similarly, if $\mu_k < 0$, then the second equality states that $\bar{d}_k$ is very close to μ_k; in fact, $n^{1/2}\bar{d}_k$ is smaller than $-n^{1/2-\epsilon}$ for any $\epsilon > 0$ and n sufficiently large.

Combined Power Gains

.508 .064 .005 .996 .958 .892 .765 .175 .018 .999 .981
.983 .531 .099 1.000 1.000 .995 .998 .753 .210 1.000 1.000

nated rejection frequencies for the six tests for SPA under the null hypothesis ($\Lambda_1 = 0$) and local alternatives ($\Lambda_1 < 0$). The rejection frequencies in italic type corres
se in regular type correspond to local powers. The reality check of White (2000) is denoted by RC_u, and the test advocated in this article is denoted by SPA_c.



4. Local Power Curves of the Four Tests, SPA_c, SPA_u, RC_c, and RC_u, for the Simulation Experiment Where $m = 100$, Λ_0

Application of RC to Technical Trading Rules

UNIVERSITY OF OXFORD

- Sullivan, Timmermann and White (1999) apply the RC to a large universe of technical trading rules
- Rules include:
  - Filter Rules
  - Moving Average Oscillators
  - Support and Resistance
  - Channel Breakout
  - On-balance Volume Averages
    - Tracks volume times return sign
    - Similar to Moving Average rules for prices
- Total of 7,846 trading rules
- Only use 1 at a time
- Use DJIA as in BLL, updated to 1996
- Consider mean return criteria and Sharpe Ratio

sample periods. Results are provided for both the Brock, Lakonishok, and LeBaron (BLL) unive
of rules. The table reports the performance measure (i.e., the annualized mean return) along w
p-value. The nominal p-value results from applying the Reality Check methodology to the bes
the data-snooping.

Sample	BLL Universe of Trading Rules			M
	Mean Return	White's p-Value	Nominal p-Value	
In-sample				
Subperiod 1 (1897–1914)	9.52	0.021	0.000	
Subperiod 2 (1915–1938)	13.90	0.000	0.000	
Subperiod 3 (1939–1962)	9.46	0.000	0.000	
Subperiod 4 (1962–1986)	7.87	0.004	0.000	
90 years (1897–1986)	10.11	0.000	0.000	
100 years (1897–1996)	9.39	0.000	0.000	
Out-of-sample				
Subperiod 5 (1987–1996)	8.63	0.154	0.055	
S&P 500 Futures (1984–1996)	4.25	0.421	0.204	

Mean Return Performance Expanded

Sample	Full Universe of Trading Rules		
	Mean Return	White's p-Value	Nominal p-Value
In-sample			
Subperiod 1 (1897–1914)	16.48	0.000	0.000
Subperiod 2 (1915–1938)	20.12	0.000	0.000
Subperiod 3 (1939–1962)	25.51	0.000	0.000
Subperiod 4 (1962–1986)	23.82	0.000	0.000
90 years (1897–1986)	18.65	0.000	0.000
100 years (1897–1996)	17.17	0.000	0.000
Out-of-sample			
Subperiod 5 (1987–1996)	14.41	0.341	0.004
S&P 500 Futures (1984–1996)	9.43	0.908	0.042

RC based on Sharpe Ratio

- From any strategy it is simple to compute the Sharpe Ratio

$$SR = \frac{P^{-1}\sum_{t=R+1}^{T}\tilde{r}_{t+1} - r_{f,t+1}}{\sqrt{P^{-1}\sum_{t=R+1}^{T}\left(\tilde{r}_{t+1} - \bar{\tilde{r}}\right)^2}}$$

- The strategy return is $\tilde{r}_{t+1} = r_{t+1}S\left(\hat{y}_{j,t+1|t}\right)$
- $\bar{\tilde{r}}$ is the mean of the strategy return
- $r_{f,t+1}$ is the risk-free rate

RC based on Sharpe Ratio

- The bootstrap can be used to compute a bootstrap version of the same rule by jointly re-sampling $\left\{\tilde{r}_{t+1}, r_{f,t+1}\right\}$
- The bootstrap Sharpe Ratio is then

$$
\begin{aligned}
SR_b^\star &= \frac{a}{\sqrt{b-c^2}} \\
a &= P^{-1}\sum_{t=R+1}^{T} \tilde{r}_{b,t+1} - r_{f,b,t+1} \\
b &= P^{-1}\sum_{t=R+1}^{T} \tilde{r}_{b,t+1}^2 \\
c &= P^{-1}\sum_{t=R+1}^{T} \tilde{r}_{b,t+1}
\end{aligned}
$$

- The SR can be computed for all models
- The RC can then be applied to the (negative) SR, rather than the (negative) return

sample periods. Results are provided for both the Brock, Lakonishok, and LeBaron (1992) (BL
universe of rules. The table reports the performance measure (i.e., the Sharpe ratio) along wi
p-value. The nominal p-value results from applying the Reality Check methodology to the bes
the data-snooping.

Sample	BLL Universe of Trading Rules			Sh
	Sharpe Ratio	White's p-Value	Nominal p-Value	
In-sample				
Subperiod 1 (1897–1914)	0.51	0.147	0.016	
Subperiod 2 (1915–1938)	0.51	0.037	0.000	
Subperiod 3 (1939–1962)	0.79	0.000	0.000	
Subperiod 4 (1962–1986)	0.53	0.051	0.003	
90 years (1897–1986)	0.45	0.000	0.000	
100 years (1897–1996)	0.39	0.000	0.000	
Out-of-sample				
Subperiod 5 (1987–1996)	0.28	0.721	0.127	
S&P 500 Futures (1984–1996)	0.23	0.702	0.165	

Sharpe Ratio Performance: Expanded

Sample	Full Universe of Trading Rules		
	Sharpe Ratio	White's p-Value	Nominal p-Value
In-sample			
Subperiod 1 (1897–1914)	1.15	0.000	0.000
Subperiod 2 (1915–1938)	0.76	0.056	0.000
Subperiod 3 (1939–1962)	2.18	0.000	0.000
Subperiod 4 (1962–1986)	1.41	0.000	0.000
90 years (1897–1986)	0.91	0.000	0.000
100 years (1897–1996)	0.82	0.000	0.000
Out-of-sample			
Subperiod 5 (1987–1996)	0.87	0.903	0.000
S&P 500 Futures (1984–1996)	0.66	0.987	0.000

Stepwise Multiple Testing

- The main issue with the Reality Check and the Test for SPA is the null
- These tests ultimately test one question:
  - Is the largest out-performance consistent with a random draw from the distribution when there are not superior models to the benchmark?
- If the null is rejected, only the best performing model can be determined to be better than the benchmark
- What about the 2nd best model? Or the k^{th} best model?
- The *StepM* extends that reality check by allowing individual models to be tested
- It is implemented by repeatedly applying a RC-like algorithm which controls the *Familywise Error Rate (FWE)*

Basic Setup

- The basic setup is identical to that of the RC/SPA
- The test is based on $\delta_{j,t} = L\left(y_{t+h}, \hat{y}_{t+h,BM|t}\right) - L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right)$
- Can be used in the same types of tests as RC/SPA
  - Absolute return
  - Sharpe Ratio
  - Risk-adjusted α comparisons
  - MSE/MAE
  - Predictive Likelihood
- Can be implemented on both raw and Studentized loss differentials

Null and Alternative Hypotheses

- The null and alternatives in StepM are not a single statement as they were in the RC/SPA
- The null**s** are

$$H_{0,j} : \mathrm{E}\left[\delta_t\right] \leq 0, \quad j = 1, \ldots, m$$

- The alternative**s** are

$$H_{1,j} : \mathrm{E}\left[\delta_t\right] > 0, \quad j = 1, \ldots, m$$

- StepM will ultimately result in a set of rejections (if any are rejected)
- Goal of StepM is to identify as many false nulls as possible while controlling the Familywise Error Rate

Familywise Error Rate

Definition (Familywise Error Rate)

For a set of null and alternative hypotheses $H_{0,i}$ and $H_{1,i}$ for $i = 1, \ldots, m$, let $\mathcal{I}_0$ contain the indices of the correct null hypotheses. The Familywise Error Rate is defined as

$$\Pr\left(\text{Rejecting at least one } H_{0,i} \text{ for } i \in \mathcal{I}_0\right) = 1 - \Pr\left(\text{Reject no } H_{0,i} \text{ for } i \in \mathcal{I}_0\right)$$

- The FWE is concerned only with the probability of making at least one Type I error
- Making 1, 2 or m Type I errors is the same to FWE
  - This is a criticism of FWE
  - Other criteria exist such as *False Discovery Rate* which controls the percentage of rejections which are false (# False Rejection/# Rejections)

Bonferroni Bounds

- Bonferoni bounds are the first procedure to control FWER

Definition (Bonferoni Bound)

Let $T_1, T_{2,} \ldots, T_m$ be a set of m test statistics, then

$$\underbrace{\Pr\left(T_1 \cup \ldots \cup T_m | H_{1,0}, \ldots H_{m,0}\right)}_{\text{Joint Probability}} \leq \sum_{j=1}^{m} \underbrace{\Pr\left(T_j | H_{0,j}\right)}_{\text{Individual Probability}}$$

where $\Pr\left(T_j | H_{0,j}\right)$ is the probability of observing $T_j$given the null $H_{0,j}$ is true.

- Bonferoni bounds are a simple method to test m hypotheses using only univariate test statistics
- Let $\left\{pv_j\right\}$ be a set of m p-values from a set of tests
- The Bonferoni bound will reject the set of nulls is $pv_j \leq \alpha/m$ for all j
  - α is the size of the test (e.g. 5%)
- When m is moderately large, this is a very conservative test
  - If $m = 5$, all $pv_j < 1\%$ to reject using 5% size
- Conservative since assumes worst case dependence among statistics

Holm's procedure

Definition (Holm's Procedure)

Let $T_1, T_{2,} \ldots, T_m$ be a set of m test statistics with associated p-values pv_j, $j = 1, \ldots, m$ where it is assumed $pv_i < pv_j$ if $i < j$. If

$$pv_j \leq \alpha / \left(m - j + 1\right)$$

then $H_{0,j}$ can be rejected in factor of $H_{1,j}$ while controlling the famliywise error rate at α.

- Example: p-values of .001, .01, .03, .05, $m = 4$, $\alpha = .05$
- Improves Bonferoni by ordering the p-values and using a stepwise procedure
- Allows subsets of hypotheses to be tested – Bonferoni is joint
- Less strict, except when $j = 1$ (same as Bonferoni)
- **Note**: Holm's procedure ends as soon as a null cannot be rejected

Relationships between testing procedures

- The RC/SPA, Bonferoni and Holm are all related

	Worst-case Dependence	Accounts for Dependence in Data
Single-step	Bonferoni	RC, SPA
Stepwise	Holm	StepM

Algorithm (StepM)

1. *Begin with the active set* $\mathcal{A} = \{1, 2, \ldots, m\}$, *superior set* $\mathcal{S} = \{\}$
2. *Construct* B *bootstraps sample* $\left\{\boldsymbol{\delta}^{\star}_{b,t}\right\}, b = 1, \ldots, B$
3. *For each bootstrap sample, compute* $T^{\star StepM}_{k,b} = \max_{j \in \mathcal{A}} \left\{\bar{\delta}^{\star}_{b,j} - \bar{\delta}_j\right\}$
4. *Compute* $q_{k,\alpha}$ *as the* $1 - \alpha$ *quantile of* $\left\{T^{\star StepM}_{k,b}\right\}$
5. *If* $\max_{j \in \mathcal{A}} \left(\bar{\delta}_j\right) < q_{k,\alpha}$ *stop*
6. *Otherwise for each* $j \in \mathcal{A}$
    a. *If* $\bar{\delta}_j \geq q_{k,\alpha}$ *add* j *to* $\mathcal{S}$ *and delete from* $\mathcal{A}$
    b. *Return to 2*

Comments

- StepM would be virtually identical to RC if only the largest $\bar{\delta}_j$ was tested
- Improves on the RC since (weakly more) individual out-performing models can be identified
- If no model outperforms, will stop with none and RC p-value will be larger than α
- Steps 2–4 are identical to the RC using the models in $\mathcal{A}$
- The stepwise testing can improve power by removing models
  - The improvement comes if a model with substantial out-performance also has large variance
  - Removing this model allows the critical value to be reduced
- StepM only guarantees that FWE$\leq \alpha$, and in general will be $< \alpha$
  - Will only $= \alpha$ if $\mathrm{E}\left[\delta_{j,t}\right] = 0$ for all j
  - Example: $N\left(\mu, \sigma^2\right)$ when $\mu < 0$, $H_0 : \mu = 0$

Studentization

- Like the SPA to the RC, the StepM can be implemented using Studentized loss differentials
- Romano & Wolf argue that the Studentization should be done *inside* each bootstrap sample, not globally as in the SPA
- Theoretically both are justified and neither makes a difference asymptotically
- Computing the variance inside each bootstrap will more closely match the re-sampled data than when using a global estimate

Algorithm (Studentized StepM)

1. *Begin with the active set* $\mathcal{A} = \{1, 2, \ldots, m\}$, *superior set* $\mathcal{S} = \{\}$
2. *Compute* $\bar{z}_j = \bar{\delta}_j / \sqrt{\hat{\omega}_j^2 / P}$ *where* $\hat{\omega}_j^2$ *was previously defined*
3. *Construct B bootstraps sample* $\left\{ \boldsymbol{\delta}_{b,t}^{\star} \right\}$, $b = 1, \ldots, B$
4. *For each bootstrap sample, compute*

$$T_{k,b}^{\star StepM} = \max_{j \in \mathcal{A}} \left\{ \frac{\bar{\delta}_{b,j}^{\star} - \bar{\delta}_j}{\hat{\omega}_j^{\star}} \right\}$$

   where $\hat{\omega}_j^{2\star}$ *is an estimate of the long-run variance of the bootstrapped data*

5. *Compute* $q_{k,\alpha}^{z}$ *as the* $1 - \alpha$ *quantile of* $\left\{ T_{k,b}^{\star StepM} \right\}$
6. *If* $\max_{j \in \mathcal{A}} \left(\bar{z}_j \right) < q_{k,\alpha}^{z}$ *stop*
7. *Otherwise for each* $j \in \mathcal{A}$
   a. *If* $\bar{z}_j \geq q_{k,\alpha}^{z}$ *add j to* $\mathcal{S}$ *and delete from* $\mathcal{A}$
   b. *Return to 2*

Why Studentization Help

- StepM is built around confidence intervals of the form

$$\left[\bar{\delta}_1 - q_{1,\alpha}, \infty\right] \times \ldots \times \left[\bar{\delta}_m - q_{1,\alpha}, \infty\right]$$

- Null hypotheses are rejected for models where 0 is *not* in its confidence interval
- In the raw form, the confidence interval is a square – the same for every loss differential
- When Studentization is used, the confidence intervals take the form

$$\left[\bar{\delta}_1 - \sqrt{\omega_1^2/P} q_{1,\alpha}^z, \infty\right] \times \ldots \times \left[\bar{\delta}_m - \sqrt{\omega_m^2/P} q_{1,\alpha}^z, \infty\right]$$

- This "customization" allows for more rejections if the loss differentials have cross-sectional heteroskedasticity

Block-size Selection

- Paper proposes a procedure to make data driven block size
- Basic idea is to use a (V)AR on $\left\{\delta_{j,t}\right\}$ to approximate the dependence
  - Similar to Den Hann-Levine HAC
- Fit AR & estimate residual covariance (or use short block bootstrap on errors)
- Simulate from model
- For $w = 1, \ldots, \overline{W}$ compute the bootstrap confidence region with size $1 - \alpha$using percentile method
- For each block size, compute the empirical coverage – percentage of simulated $\bar{\delta}$ in their confidence region
- Choose optimal w which most closely matches $1 - \alpha$
  - Alternative: Use Politis & White

Empirical Application

- Applied StepM to a set of 105 Hedge Fund Returns with long histories
- Returns net of management fees
- Benchmark model was *risk-free rate*
- $m = 105, P = 147$ (all out-of-sample)
- Results:
  - Raw data: No out-performers
    - Max ratio of standard deviation $\hat{\omega}_i/\hat{\omega}_j = 22$
  - Studentized: 7 funds identified
- **Note**: Will *always* identify funds with the largest $\bar{\delta}$ (or $\bar{z}$) first

TABLE VIII

THE TEN LARGEST BASIC AND STUDENTIZED TEST STATISTICS, TOGETHER WITH THE CORRESPONDING HEDGE FUNDS, IN OUR EMPIRICAL APPLICATION

$\bar{x}_{T,s} - \bar{x}_{T,S+1}$	Fund	$(\bar{x}_{T,s} - \bar{x}_{T,S+1})/\hat{\sigma}_{T,s}$	Fund
1.70	Libra Fund	10.63	Market Neutral*
1.41	Private Investment Fund	9.26	Market Neutral Arbitrage*
1.36	Aggressive Appreciation	8.43	Univest (B)*
1.27	Gamut Investments	6.33	TQA Arbitrage Fund*
1.26	Turnberry Capital	5.48	Event-Driven Risk Arbitrage*
1.14	FBR Weston	5.29	Gabelli Associates*
1.11	Berkshire Partnership	5.24	Elliott Associates**
1.09	Eagle Capital	5.11	Event Driven Median
1.07	York Capital	4.97	Halcyon Fund
1.07	Gabelli Intl.	4.65	Mesirow Arbitrage Trust

[a]The return unit is 1%. Funds identified in the first step are indicated by the superscript * and funds identified in the second step are indicated by the superscript **.

[27]The risk-free rate is a simple and widely accepted benchmark. Of course, our methods also apply to alternative benchmarks such as hedge fund indices or multifactor hedge fund benchmarks; for example, see Kosowski, Naik, and Teo (2005).

[28]To account for leftover dependence not captured by the VAR(1) model, we use the stationary bootstrap with average block size $b = 5$ to bootstrap the residuals.

Improving StepM using SPA

- The main step in the StepM algorithm is identical to the RC
- The important difference is that the test is implemented for each null, rather than globally
- StepM will suffer if very poor models are included with a large variance
  - Especially true for raw version, but also relevant for Studentized version
  - Example

$$\begin{bmatrix} \bar{\delta}_1 \\ \bar{\delta}_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ -5 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)$$

  - Reality Check critical value will be 1.95, while "best" critical value would be 1.645 (since only 1 relevant for asymptotic distribution)
- The RC portions of StepM can be replaced by SPA versions which addresses this problem
- Simple as adding in the indicator function I_j^c when subtracting the mean in step 3 (step 4 in Studentized version)
- Using SPA modification will always find more out-performing models

Model Confidence Set (MCS)

UNIVERSITY OF OXFORD

- RC, SPA and StepM were all testing superior predictive ability
- This type hypothesis is common when there is a natural benchmark
- In some scenarios there may not be a single benchmark, or there may more than one models which could be considered benchmarks
- When this occurs, it is not clear
  - How to implement RC/SPA/StepM
  - How to make sound conclusions about superior predictive ability
- The model confidence set addresses this problem by *bypassing the benchmark*
- The MCS aims to find the *best model* and all models which are *indistinguishable from the best*
  - The model with the lowest loss will always be the best – identifying the others is more challenging
- Also returns p-values for models with respect to the MCS

Notation Preliminaries

- The outcome of the MCS is a *set of models*
  - All model sets will be denoted using $\mathcal{M}$
- The initial model set is $\mathcal{M}_0$
- The goal is to find $\mathcal{M}^\star$ which is the set of all models which are indistinguishable from the best
- The output of the MCS algorithm is $\widehat{\mathcal{M}}_{1-\alpha}$ where α is the size of the test
  - The size is interpreted as a Familywise Error Rate – same as StepM
  - In general $\widehat{\mathcal{M}}_{1-\alpha}$ will contain more than 1 model
- In between $\mathcal{M}_0$ and $\widehat{\mathcal{M}}_{1-\alpha}$ are other sets of models

$$\mathcal{M}_0 \supset \mathcal{M}_1 \supset \ldots \supset \widehat{\mathcal{M}}_{1-\alpha}$$

Notation Preliminaries

- To construct the model confidence set, two tools are needed
  - An equivalence test $d_{\mathcal{M}}$: Determines whether the model in $\mathcal{M}$ are equal in terms of loss
  - An elimination rule $e_{\mathcal{M}}$: Determines which model to eliminate if $d_{\mathcal{M}}$ finds that the models are not equivalent
- The generic form of the algorithm, starting at $i = 0$:
  1. Apply $d_{\mathcal{M}}$ to $\mathcal{M}_i$
  2. If $d_{\mathcal{M}}$ rejects equivalence, use $e_{\mathcal{M}}$to eliminate 1 model to produce $\mathcal{M}_{i+1}$
     a. If not, stop
  3. Increment i, return to 1
- Has a similar flavor to StepM
  - Also gains from eliminating models with high variance

The Model Confidence Set

- When the algorithm ends, the final set $\widehat{\mathcal{M}}_{1-\alpha}$ has the property

$$\lim_{P\to\infty} \Pr\left(\mathcal{M}^\star \subset \widehat{\mathcal{M}}_{1-\alpha}\right) \geq 1-\alpha$$

- The result follows directly since the FWE is $\leq \alpha$
- If there is only 1 "best" model, then the result can be strengthened

$$\lim_{P\to\infty} \Pr\left(\mathcal{M}^\star \subset \widehat{\mathcal{M}}_{1-\alpha}\right) = 1$$

  - The MCS will find the "best" model asymptotically
  - The intuition behind this is that the "best" model will have:
    - Lower loss than all other models
    - The variance of the average loss differential will decline as $P \to \infty$
- When 2 or more models are equally good, there is always a α chance that at least 1 will be rejected
- In large samples, models which are not in $\mathcal{M}^\star$ will be eliminated with probability 1 since the individual test statistics are consistent

Model Confidence Set

- The MCS takes loss functions as inputs, but ultimately works on loss differentials
- Since there is no benchmark model, all loss differentials are considered

$$\delta_{ij,t} = L\left(y_{t+h}, \hat{y}_{t+h,i|t}\right) - L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right)$$

- There are many pairs, and so the actual test examines whether the average loss for model j is different from that of all models

$$\bar{\delta}_i = \frac{1}{m-1} \sum_{i=1, i \neq j}^{m} \bar{\delta}_{ij}$$

- If $\bar{\delta}_i$ is sufficiently positive, then model i is worse then the other models in the set

Null and Alternative

- The MCS can be based on two test statistics
- Both satisfy some technical conditions on $d_{\mathcal{M}}$ and $e_{\mathcal{M}}$
- The first is based on $T = \max_{i \in \mathcal{M}} (\bar{z}_i)$ where $\bar{z}_i = \bar{\delta}_i / \hat{\sigma}_i$ and $\hat{\sigma}_i^2$ is an estimate of the (log-run) variance of $\bar{\delta}_i$
  - The elimination rule is $e_{\mathcal{M}} = \operatorname{argmax}_{i \in \mathcal{M}} z_i$
- The second is based on $T_R = \max_{i,j \in \mathcal{M}} \left| \bar{z}_{ij} \right|$ where $\bar{z}_{ij} = \bar{\delta}_{ij} / \hat{\sigma}_{ij}$ and $\hat{\sigma}_{ij}$ is an estimate of the (log-run) variance of $\bar{\delta}_{ij}$
  - The elimination rule is $e_{R,\mathcal{M}} = \operatorname{argmax}_{i \in \mathcal{M}} \sup_{j \in \mathcal{M}} \bar{z}_{ij}$
  - Eliminate the model which has the largest loss differential to some other model, relative to its standard deviation
- At each step the null is $H_0 : \mathcal{M} = \mathcal{M}^\star$ and the alternative is $H_1 : \mathcal{M} \supsetneq \mathcal{M}^\star$

Model Confidence Set Setup

Algorithm (Model Confidence Set Components)

1. *Construct a set of bootstrap indices which will be reused throughout the MCS construction using a bootstrap appropriate for the data*
2. *Construct the average loss for each model*

$$\bar{L}_j = P^{-1} \sum_{t=R+1}^{T} L_{j,t}$$

where $L_{j,t} = L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right)$

3. *For each bootstrap replication, compute centered the bootstrap average loss*

$$\eta^{\star}_{b,j} = P^{-1} \sum_{t=R+1}^{T} L^{*}_{b,j,t} - \bar{L}_j$$

UNIVERSITY OF OXFORD

Algorithm (Model Confidence Set)

1. *Being with $\mathcal{M} = \mathcal{M}_0$ containing all models where m is the number of models in $\mathcal{M}$*
2. *Calculate $\bar{L} = m^{-1} \sum_{j=1}^{m} \bar{L}_j$, $\eta_b^\star = m^{-1} \sum_{j=1}^{m} \eta_{b,j}^\star$, and $\hat{\sigma}_j^2 = B^{-1} \sum_{b=1}^{B} \left(\eta_{b,j}^\star - \bar{\eta}_j^\star \right)^2$ where $\bar{\eta}_j^\star$ is the average of $\eta_{b,j}^*$ for model j*
3. *Define $T = \max_{j \in \mathcal{M}} \left(\bar{z}_j \right)$ where $\bar{z}_j = \bar{L}_j / \hat{\sigma}_j$*
4. *For each bootstrap sample, compute $T_b^\star = \max_{j \in \mathcal{M}} \left(\left(\bar{L}_{b,j}^\star - \bar{L}_b^\star \right) / \hat{\sigma}_j \right) = \max_{j \in \mathcal{M}} \left(\left(\eta_{b,j}^\star - \eta_b^\star \right) / \hat{\sigma}_j \right)$*
5. *Compute the p-value of $\mathcal{M}$ as $\hat{p} = B^{-1} \sum_{b=1}^{B} I \left[T_b^\star > T \right]$*
6. *If $\hat{p} > \alpha$ stop*
7. *If $\hat{p} < \alpha$, set $e_{\mathcal{M}} = \operatorname{argmax}_{j \in \mathcal{M}} \left(\bar{z}_j \right)$ and eliminate the model with the largest test statistic from $\mathcal{M}$*
8. *Return to step 2, using the reduced model set*

Comments

- It is important that the variance estimates are re-computed in each step of algorithm
- This allows the standard errors to decline if poor models are excluded since the cross-sectional variance of $\bar{L}_j$ should be smaller when a bad model is dropped
- In practice the MCS should be implemented by computing in order
  1. A set of bootstrap indices
  2. The P by m set of bootstrapped losses $L^*_{b,j,t}$
  3. The 1 by m vector containing $\eta^\star_{b,j}$
- By iterating over these B times only the B by m matrix containing $\eta^\star_{b,j}$ has to be retained
  - Plus the 1 by m vector containing $\bar{L}_j$

Model Confidence P-value

- The MCS can also provide p-values for each model
- If model i is eliminated, then the p-value of model i is the maximum of the $\hat{p}$ found when model i is eliminated and *all previous p-values*
- Suppose $\alpha = .05$, and the first three rounds eliminated models with $\hat{p}$ of .01,.04,.02, respectively
- The three p-values would then be:
  - $.01$(nothing to compare against)
  - $.04 = \max(.01, .04)$
  - $.04 = \max(.02, .04)$
- The output of the MCS algorithm is $\widehat{\mathcal{M}}_{1-\alpha}$ which contains the true set of best models with probability weakly larger than $1 - \alpha$
- This is similar to a standard frequentist confidence interval which contains the true parameter with probability of at least $1 - \alpha$
- The MCS p-value is not a statement about the probability that a model is the best
  - For example, the model with the lowest loss always has p-value = 1

Then for some $j \leq k$ we have $P_{H_{0,\mathcal{M}_j}} \geq \alpha$, in which case $H_{0,\mathcal{M}_j}$ is accepted at significance level α which terminates the MCS procedure before the elimination rule gets to $e_{\mathcal{M}_k} = i$. So $\hat{p}_i \geq \alpha$ implies $i \in \widehat{\mathcal{M}}^*_{1-\alpha}$.
This completes the proof. ■

Table 1: Computation of MCS p-values

Elimination Rule	p-value for $H_{0,\mathcal{M}_k}$	MCS p-value
$e_{\mathcal{M}_1}$	$P_{H_{0,\mathcal{M}_1}} = 0.01$	$\hat{p}_{e_{\mathcal{M}_1}} = 0.01$
$e_{\mathcal{M}_2}$	$P_{H_{0,\mathcal{M}_2}} = 0.04$	$\hat{p}_{e_{\mathcal{M}_2}} = 0.04$
$e_{\mathcal{M}_3}$	$P_{H_{0,\mathcal{M}_3}} = 0.02$	$\hat{p}_{e_{\mathcal{M}_3}} = 0.04$
$e_{\mathcal{M}_4}$	$P_{H_{0,\mathcal{M}_4}} = 0.03$	$\hat{p}_{e_{\mathcal{M}_4}} = 0.04$
$e_{\mathcal{M}_5}$	$P_{H_{0,\mathcal{M}_5}} = 0.07$	$\hat{p}_{e_{\mathcal{M}_5}} = 0.07$
$e_{\mathcal{M}_6}$	$P_{H_{0,\mathcal{M}_6}} = 0.04$	$\hat{p}_{e_{\mathcal{M}_6}} = 0.07$
$e_{\mathcal{M}_7}$	$P_{H_{0,\mathcal{M}_7}} = 0.11$	$\hat{p}_{e_{\mathcal{M}_7}} = 0.11$
$e_{\mathcal{M}_8}$	$P_{H_{0,\mathcal{M}_8}} = 0.25$	$\hat{p}_{e_{\mathcal{M}_8}} = 0.25$
$\vdots$	$\vdots$	$\vdots$
$e_{\mathcal{M}_{(m_0)}}$	$P_{H_{0,\mathcal{M}_{m_0}}} \equiv 1.00$	$\hat{p}_{e_{\mathcal{M}_{m_0}}} = 1.00$

The table illustrates the computation of MCS p-values. Note that MCS p-values for some models do not coincide with the p-values for the corresponding null hypotheses. For example, the MCS p-value for $e_{\mathcal{M}_3}$ (the third model to be eliminated) exceeds the p-value for $H_{0,\mathcal{M}_3}$ because the p-value associated with $H_{0,\mathcal{M}_2}$ – a null hypothesis tested prior to $H_{0,\mathcal{M}_3}$ – is larger.

The interpretation of a MCS p-value is analogous to that of a classical p-value. The analogy is to a $(1-\alpha)$ confidence interval that contains the 'true' parameter with a probability no less than $1-\alpha$. The MCS

Model Confidence Set using T_R

Algorithm (Model Confidence Set Components)

1. *Construct a set of bootstrap indices which will be reused throughout the MCS construction using a bootstrap appropriate for the data*
2. *Construct the average loss for each model* $\bar{L}_j = P^{-1} \sum_{t=R+1}^{T} L_{j,t}$ *where* $L_{j,t} = L\left(y_{t+h}, \hat{y}_{t+h,j|t}\right)$
3. *For each bootstrap replication, compute centered the bootstrap average loss*

$$\bar{L}^{\star}_{b,j} = P^{-1} \sum_{t=R+1}^{T} L^{*}_{b,j,t} - \bar{L}_j$$

4. *Calculate*

$$\hat{\sigma}^2_{ij} = B^{-1} \sum_{b=1}^{B} \left(\left(\bar{L}^{\star}_{b,i} - \bar{L}^{\star}_{i} \right) - \left(\bar{L}^{\star}_{b,j} - \bar{L}^{\star}_{j} \right) \right)^2$$

where $\bar{L}^{\star}_{j}$ *is the average of* $\bar{L}^{\star}_{b,j}$ *for the model* j *across all bootstraps*

UNIVERSITY OF OXFORD

Algorithm (Model Confidence Set)

1. *Being with $\mathcal{M} = \mathcal{M}_0$ containing all models where m is the number of models in $\mathcal{M}$*
2. *Define $T_R = \max_{i,j \in \mathcal{M}} \left(\bar{z}_{ij} \right)$ where $\bar{z}_{ij} = \left| \bar{L}_i - \bar{L}_j \right| / \hat{\sigma}_{ij}$*
3. *For each bootstrap sample, compute $T^{\star}_{R,b} = \max_{i,j \in \mathcal{M}} \left(\left| \bar{L}^{\star}_i - \bar{L}^{\star}_j \right| / \hat{\sigma}_{ij} \right)$*
4. *Compute the p-value of $\mathcal{M}$ as*

$$\hat{p} = B^{-1} \sum_{b=1}^{B} I \left[T^{\star}_{R,b} > T_R \right]$$

5. *If $\hat{p} > \alpha$ stop*
6. *If $\hat{p} < \alpha$, set $e_{\mathcal{M}} = \operatorname{argmax}_{i \in \mathcal{M}} \sup_{j \in \mathcal{M}} \left(\bar{z}_{ij} \right)$ and eliminate the model with the largest test statistic from $\mathcal{M}$*
7. *Return to step 2, using the reduced model set*

Comments

- The main difference is that the variance is *not* re-estimated in each iteration
- This happens since T_R is based on the maximum DMW test statistic in each iteration
  - DMW only depends on the properties of the pair
- However, the bootstrapped distribution does depend on which models are included and so this will vary across the iterations
- This version of the algorithm requires storing the B by m matrix of $\bar{L}_j^\star$

Confidence sets for ICs

- The MCS can be used to construct confidence sets for ICs
- This type of comparison does not directly use forecasts, and so is in-sample
- This differs from traditional model selection where only the model with the best IC is chosen
- The MCS for an IC could be used as a pre-filtering mechanism prior to combining
- Implementing the MCS on an IC is slightly more complicated than the default MCS since it is necessary to jointly bootstrap the vector $\left\{y_t, \mathbf{x}_{j,t}\right\}$ where $\mathbf{x}_{j,t}$ are the regressors in model j
- Paper recommends using T_R statistic to compare models using IC
- The object of interest is

$$IC_j = T \ln \hat{\sigma}_j^2 + c_j$$

- c_j is the penalty term
  - AIC: $2k_j$, BIC: $k_j \ln T$
  - AIC*: $2k_j^\star$, BIC*: $k_j^\star \ln T$
- $k_j^\star$ is known as *effective degrees of freedom* (in mis-specified model $k^\star \neq k$)
- MCS paper discusses how to estimate $k^\star$

Confidence sets for ICs

- Using T_R MCS construction algorithm, the test statistic is based on

$$T_R = \max_{i,j \in \mathcal{M}} \left| \left[T \ln \hat{\sigma}_i^2 + c_i \right] - \left[T \ln \hat{\sigma}_j^2 + c_j \right] \right|$$

- The bootstrap critical values are computed from

$$T^{\star}_{R,b} = \max_{i,j \in \mathcal{M}} \left(\left[T \ln \hat{\sigma}_i^{2\star} + c_i - T \ln \hat{\sigma}_i^2 \right] - \left[T \ln \hat{\sigma}_j^{2\star} + c_j - T \ln \hat{\sigma}_j^2 \right] \right)$$

- $\hat{\sigma}_i^{2\star}$ is the variance computed using

$$\epsilon^{\star}_{b,t} = y^{\star}_{b,t} - \mathbf{x}^{\star\prime}_{b,j,t} \hat{\boldsymbol{\beta}}^{\star}_{b,j}$$

- $\hat{\boldsymbol{\beta}}^{\star}_{b,j}$ is re-estimated using the bootstrapped data $\left\{ y^{\star}_{b,t}, \mathbf{x}^{\star}_{b,j,t} \right\}$
- Errors are computed using the bootstrapped data and parameter estimates
- Aside from these changes, the remainder of the algorithm is unmodified

False Discovery Rate and FWER

- Controlling False Discover Rate (FDR) is an alternative to controlling Family Wise Error Rate (FWER)

Definition (k-Familywise Error Rate)

For a set of null and alternative hypotheses $H_{0,i}$ and $H_{1,i}$ for $i = 1, \ldots, m$, let $\mathcal{I}_0$ contain the indices of the correct null hypotheses. The k-Familywise Error Rate is defined as

$$\Pr\left(\text{Rejecting at least } k\ H_{0,i} \text{ for } i \in \mathcal{I}_0\right) = 1 - \Pr\left(\text{Reject no } H_{0,i} \text{ for } i \in \mathcal{I}_0\right)$$

- k is typically 1, so the testing procedures control the probability of any number of false rejections
  - Type I errors
- The makes FWER tests possibly conservative
  - Depends on what the actual intent of the study is

False Discovery Rate

Definition

The False Discovery Rate is the percentage of false null hypothesis relative to the total number of rejections, and is defined

$$FDR = {}^{F}/_{R}$$

where F is the number of false rejections and R is the total number of rejections.

- Unlike FWER, methods that control FDR explicitly assume that some rejections are false.
- Ultimately this leads to a (potentially) procedure that might discover more actual rejections
- For standard DMW-type tests, both FWER and FDR control fundamentally reduce to choosing a critical value different from the usual ± 1.96
  - Most of the time larger in magnitude
  - Can be smaller in the case of FDR when there are many false nulls

False Discovery Rate

UNIVERSITY OF OXFORD

- FDR is naturally *adaptive*
- When the number of false nulls is small (~0), then FDR should choose a critical value similar to the FWER-based procedures
  - $R \approx F, F/R \approx 1$ so any F is too large
  - On the other hand, when the percentage of false nulls is near 100%, can reject all nulls
    - $F \approx 0, F/R \approx 0$ and all nulls can be rejected
    - Critical value can be arbitrarily small since virtually no tests have small values
    - Hypothetically, could have a critical value of 0 if all nulls were actually false
- FDR controls the false rejection rate, and it is common to use rates in the range of 5-10%
  - Ultimately should depend on risk associated with trading a bad strategy against the cost of missing a good strategy
  - Adding a small percentage of near 0 excess return strategies to a large set of useful strategies shouldn't deteriorate performance substantially

Operationalizing FDR

- Operationalizing FDR requires some estimates
- In standard trading strategy setup, $H_0 : \mu = 0, H_A : \mu \neq 0$ where μ is the expected return in excess of some benchmark
  - Benchmark might be risk-free rate, or could be buy-and-hold strategy
- π is the proportion of false nulls
  - Estimated using information about the distribution of p-values "near" 1 since these should all be generated from true nulls
  - Entire procedure relies on only p-values
    - Similar to Bonferoni or Bonferoni-Holm
  - For standard 2-sided alternative

$$p_i = 2\left(1 - \Phi\left(|t_i|\right)\right)$$

where t_i is (normalized) test statistic for strategy i.

Computing FDR

- Key idea is to find γ, which is some number in $[0, 1]$ such that

$$\alpha = \widehat{FDR} \equiv \frac{\hat{\pi} l \gamma}{\sum_{i=1}^{l} I\left[p_i < \gamma\right]}$$

- where
  - α is the target FDR rate
  - $\hat{\pi}$ and an estimate of the percentage of nulls that are true (no abnormal performance)
  - l is the number of rules
  - γ is the parameter that is used to find the p-value cutoff
  - $\sum_{i=1}^{l} I\left[p_i < \gamma\right]$ is the number of rejections using γ
- The numerator is simply an estimate of the number of false rejections, which is
  Probability of Null True $\times$ Number of Hypotheses = Number of True Hypotheses
  Number of False Hypotheses $\times$ Cutoff = Number of False that are Rejected using γ
- Exploits the fact that under the null p-values have a uniform distribution, so that if there are M false nulls, then, using a threshold of γ will reject γM

Positive and Negative FDR

- Can further decompose FDR into upper (better) and lower (worse) measures

$$\widehat{FDR}^{+} \equiv \frac{{}^{1}/_{2}\hat{\pi}_{l}\gamma_{U}}{\sum_{i=1}^{l} I\left[p_i < \gamma_U, t_i > 0\right]}, \quad \widehat{FDR}^{+} \equiv \frac{{}^{1}/_{2}\hat{\pi}_{l}\gamma_{L}}{\sum_{i=1}^{l} I\left[p_i < \gamma_L, t_i < 0\right]}$$

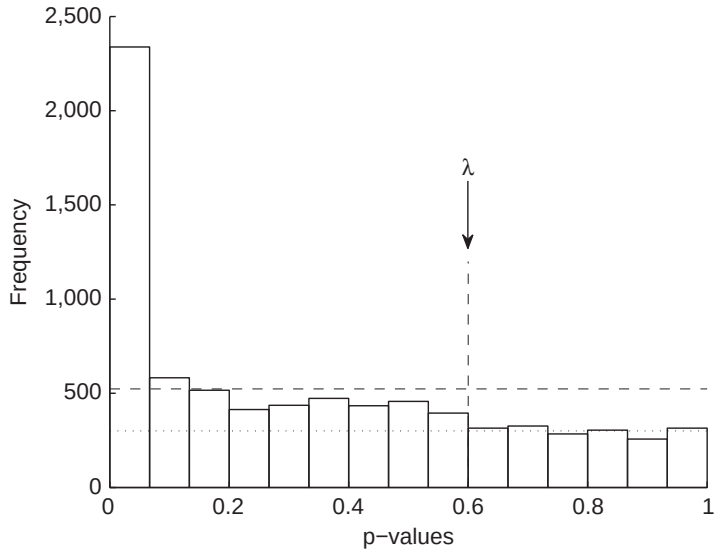
- This version assumes a symmetric 2-sided test statistic, so that on average 50% of the false rejections are in each tail
- Allows for tail-specific choice of γ which would naturally vary if the number of correct rejections was different
  - Suppose for example that many rules were bad, then γ_L would be relatively large

Estimation of π

- π is estimated as

$$\hat{\pi} = \frac{\sum_{i=1}^{l} I\left[p_k > \lambda\right]}{l(1-\lambda)}$$

- $\lambda$is a tuning parameter
  - Simple to choose using visual inspection
  - Recall that true nulls lead to a flat p-value histogram
  - Find point where histogram looks non-flat, use cutoff for λ
- Histogram from BS



Estimating π

- $\hat{\pi}$ allows percentage of correct rejections to be computed as $\hat{\pi}^A = 1 - \hat{\pi}$
- In the decomposed FDR the number of good (bad) rules can be computed as

$$\alpha \times \sum_{i=1}^{l} I\left[p_i < \gamma_U, t_i > 0\right]$$

  - Note that γ_U is fixed here

Bajgrowicz & Scaillet (*JFE*, 2012)

UNIVERSITY OF OXFORD

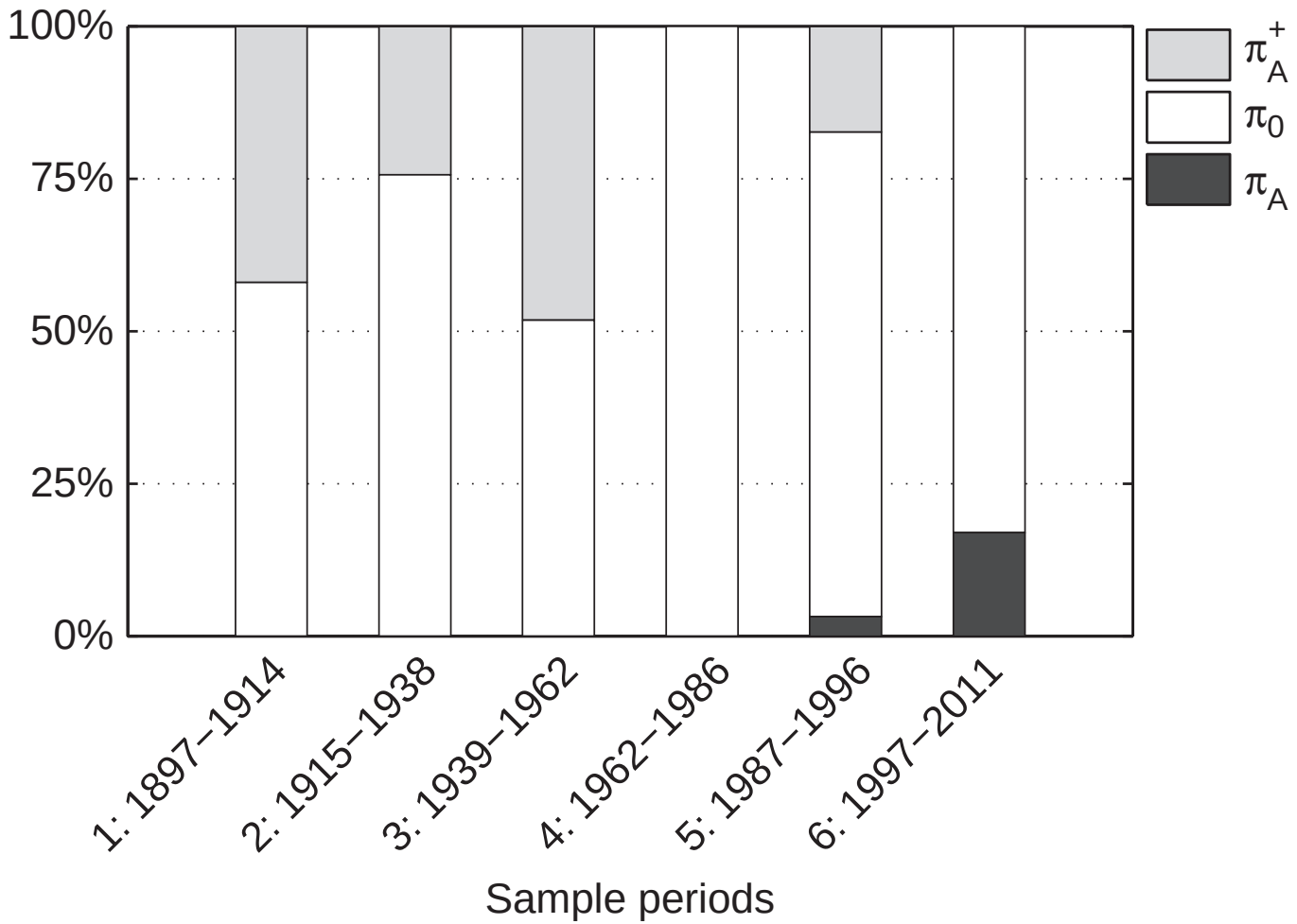
- Apply FDR to technical trading rules of STW
- Use DJIA
  - 1897-2011
- Find similar results, although importantly consider transaction costs for break even
  - Strategies that trade more can have higher means while not violating EMH

Background on Competitor Methods

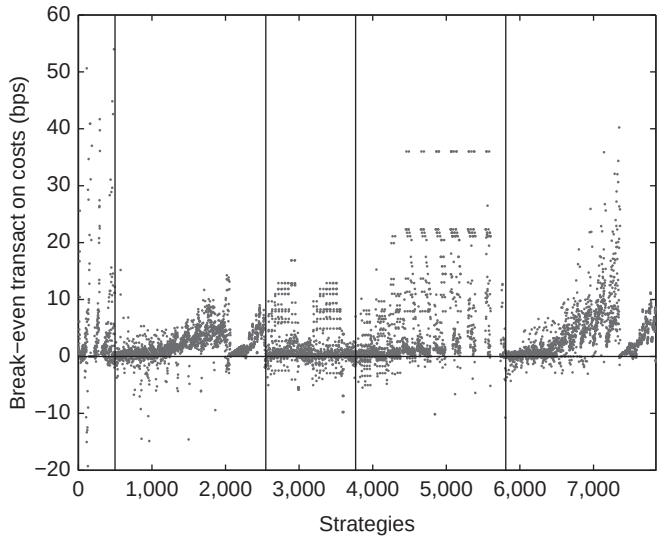
UNIVERSITY OF OXFORD

Sample period	RW portfolio		Best rule		DJIA
	Sharpe ratio	Portfolio size	Sharpe ratio	BRC *p*-value	Sharpe ratio
1: 1897–1914	1.24	45	1.18	0.00	−0.12
2: 1915–1938	–	0	0.73	0.11	0.06
3: 1939–1962	1.49	62	2.34	0.00	0.41
4: 1962–1986	1.52	15	1.45	0.00	−0.16
5: 1987–1996	–	0	0.84	0.93	0.66
6: 1997–2011	–	0	0.48	1.00	0.12
1897–1996	0.70	88	0.82	0.00	0.12

Good and Bad Rules



Transaction Costs Required for 0-profit (-1962)

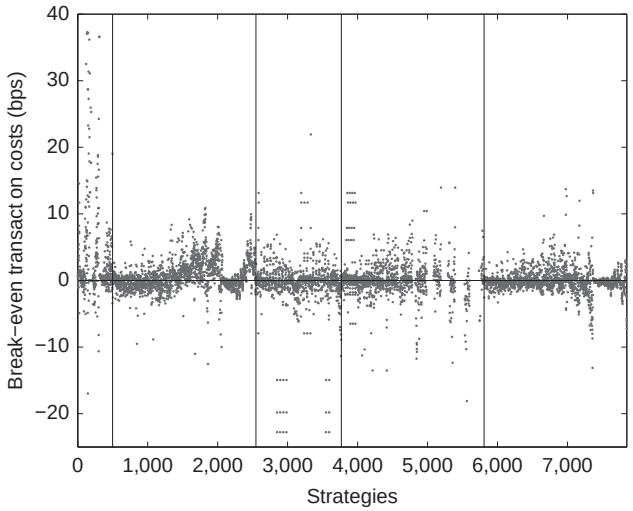


- Transaction costs are important when assessing rules
- Rather than apply arbitrary TC, look for break even
- Transaction costs are a function of mean and number of transactions

$$0 = \mu_i - TC \times \#\{trades\}$$

- μ_i is the full-sample mean, not the annualized

Transaction Costs Required for 0-profit (1962-)



- Transaction for break even are lower
- Actual transaction costs are lower
- Unclear whether this is driven by more trading signals or worse mean

Persistence of Rules

Sample period	FDR portfolio			RW portfolio			50 best rules		Best rule	
	IS	OOS	Median size	IS	OOS	Median size	IS	OOS	IS	OOS
1: 1897–1914	3.41	0.47	14	1.31	0.51	0	5.79	0.50	6.34	0.03
2: 1915–1938	4.62	0.01	13	0.90	0.17	0	5.39	−0.03	5.98	0.09
3: 1939–1962	4.77	0.55	15	1.85	0.09	0	5.78	0.43	6.70	0.12
4: 1962–1986	5.34	−0.31	13	1.36	0.14	0	6.17	−0.18	6.95	−0.59
5: 1987–1996	4.52	−0.34	12	–	–	–	5.44	−0.37	6.07	0.08
6: 1997–2011	4.55	−0.74	12	0.78	0.07	0	5.22	−0.51	5.97	−0.27

- Sharpe-Ratios
- Persistence is low
- Conservative Romano-Wolf appears to have more persistence
- Combination appears to be not help