Fast orthogonal transforms and generation of Brownian paths

G. Leobacher

Summer 2011

Discrete Brownian paths

What - and why

Many financial derivatives can be priced using

$$\text{price} = E\left(f(B)\right)$$

where B is a standard Brownian motion, $B = (B_t)_{t\in[0,1]}$ (Black-Scholes formula) or

$$\text{price} = E\left(f(B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})\right)$$

and often the second formula is taken as an approximation for the first one.

Discrete Brownian paths

What - and why

Many financial derivatives can be priced using

$$\text{price} = E\left(f(B)\right)$$

where B is a standard Brownian motion, $B = (B_t)_{t\in[0,1]}$ (Black-Scholes formula) or

$$\text{price} = E\left(f(B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})\right)$$

and often the second formula is taken as an approximation for the first one.

Discrete Brownian paths

What - and why

Many financial derivatives can be priced using

$$\text{price} = E\left(f(B)\right)$$

where B is a standard Brownian motion, $B = (B_t)_{t\in[0,1]}$ (Black-Scholes formula) or

$$\text{price} = E\left(f(B_{\frac{1}{n}},\ldots,B_{\frac{n}{n}})\right)$$

and often the second formula is taken as an approximation for the first one.

Discrete Brownian paths

What - and why

Many financial derivatives can be priced using

$$\text{price} = E\left(f(B)\right)$$

where B is a standard Brownian motion, $B = (B_t)_{t\in[0,1]}$ (Black-Scholes formula) or

$$\text{price} = E\left(f(B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})\right)$$

and often the second formula is taken as an approximation for the first one.

Discrete Brownian paths

What - and why

We therefore want efficient ways to generate a random vector

$$B = (B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})$$

where

$$B_{\frac{1}{n}}, B_{\frac{2}{n}} - B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}} - B_{\frac{n-1}{n}}$$

are independent normal variables with mean 0 and variance $\frac{1}{n}$, from

$$X_1, \ldots, X_n \ldots i.i.d. N(0, 1)$$

We therefore want efficient ways to generate a random vector

$$B = (B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})$$

where

$$B_{\frac{1}{n}}, B_{\frac{2}{n}} - B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}} - B_{\frac{n-1}{n}}$$

are independent normal variables with mean 0 and variance $\frac{1}{n}$, from

$$X_1, \ldots, X_n \ldots i.i.d. N(0,1)$$

Discrete Brownian paths

What - and why

We therefore want efficient ways to generate a random vector

$$B = (B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})$$

where

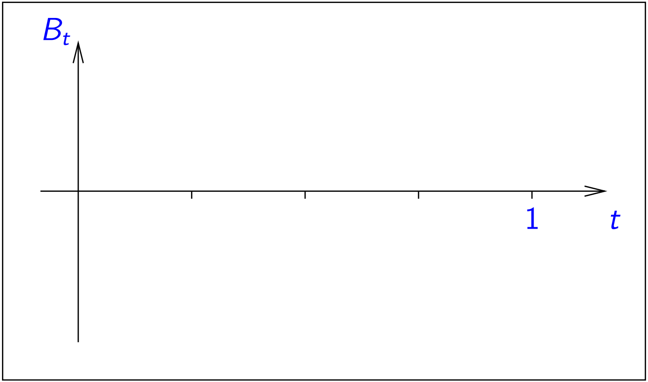
$$B_{\frac{1}{n}}, B_{\frac{2}{n}} - B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}} - B_{\frac{n-1}{n}}$$

are independent normal variables with mean 0 and variance $\frac{1}{n}$, from

$$X_1, \ldots, X_n \ldots i.i.d. N(0,1)$$

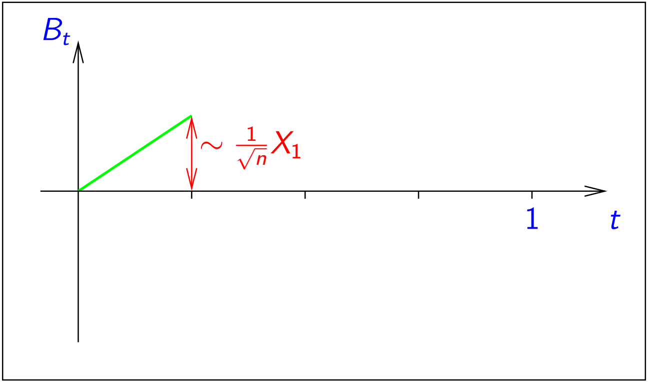
Forward construction

a.k.a. random walk construction, step-by-step method, crude method



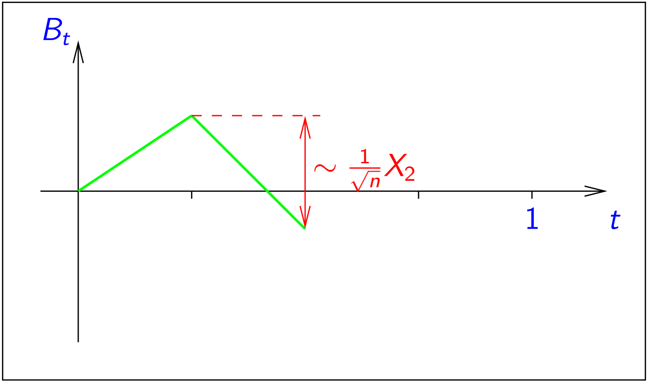
Forward construction

a.k.a. random walk construction, step-by-step method, crude method



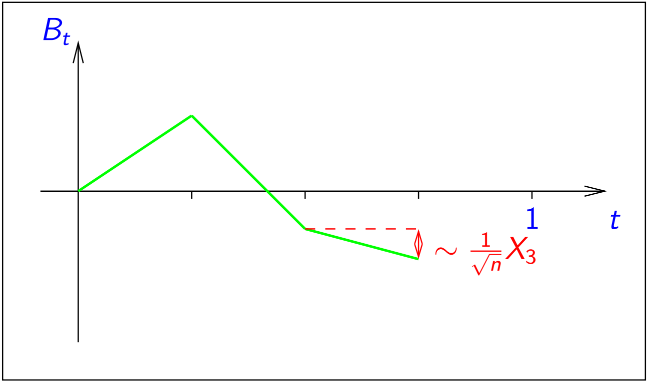
Forward construction

a.k.a. random walk construction, step-by-step method, crude method



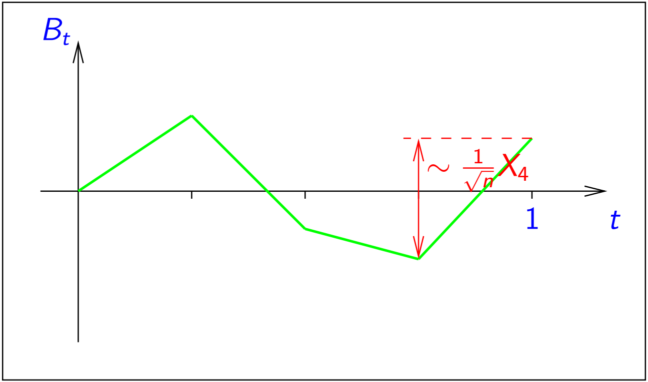
Forward construction

a.k.a. random walk construction, step-by-step method, crude method



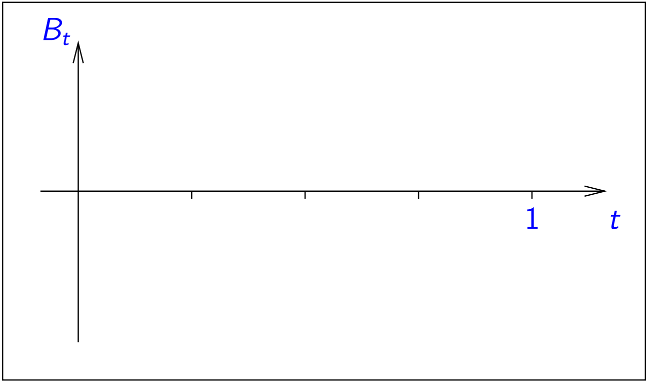
Forward construction

a.k.a. random walk construction, step-by-step method, crude method



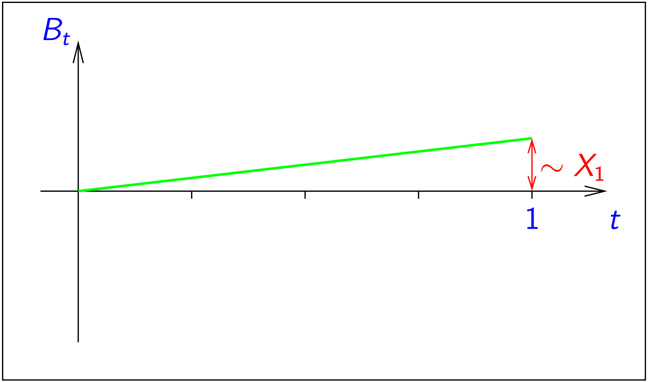
Brownian bridge construction

a.k.a. Lévy-Ciesielski construction, midpoint displacement



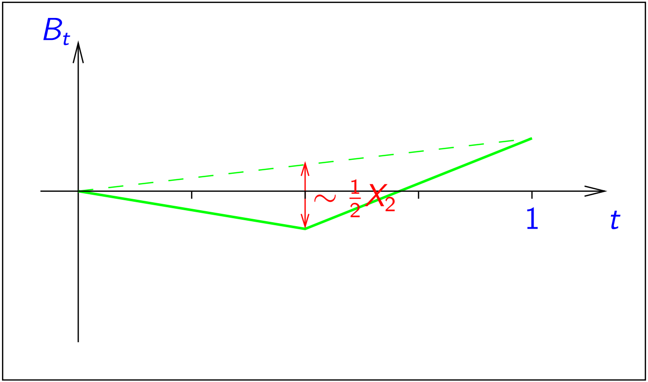
Brownian bridge construction

a.k.a. Lévy-Ciesielski construction, midpoint displacement



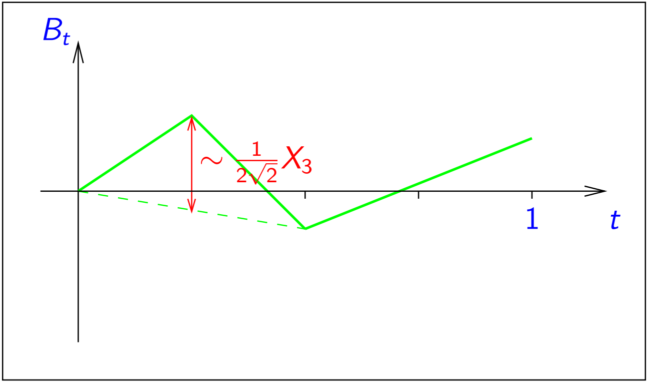
Brownian bridge construction

a.k.a. Lévy-Ciesielski construction, midpoint displacement



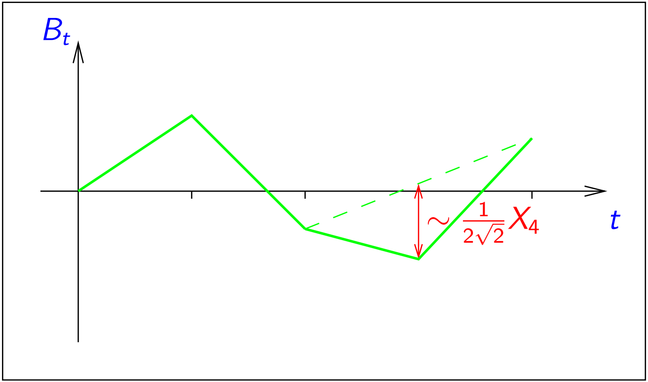
Brownian bridge construction

a.k.a. Lévy-Ciesielski construction, midpoint displacement



Brownian bridge construction

a.k.a. Lévy-Ciesielski construction, midpoint displacement



Brownian bridge construction

- Probabilistically equivalent to forward construction
- Influence of X_k on overall behavior of B is decreasing with k
  - stratified sampling
  - quasi-Monte Carlo
- Cost of generating path on n nodes is of same order as for forward construction: $O(n)$

First application in financial/QMC context: Moskowitz & Caflisch (1996)

Brownian bridge construction

- Probabilistically equivalent to forward construction
- Influence of X_k on overall behavior of B is decreasing with k
  - stratified sampling
  - quasi-Monte Carlo
- Cost of generating path on n nodes is of same order as for forward construction: $O(n)$

First application in financial/QMC context: Moskowitz & Caflisch (1996)

Brownian bridge construction

- Probabilistically equivalent to forward construction
- Influence of X_k on overall behavior of B is decreasing with k
  - stratified sampling
  - quasi-Monte Carlo
- Cost of generating path on n nodes is of same order as for forward construction: $O(n)$

First application in financial/QMC context: Moskowitz & Caflisch (1996)

Brownian bridge construction

- Probabilistically equivalent to forward construction
- Influence of X_k on overall behavior of B is decreasing with k
  - stratified sampling
  - quasi-Monte Carlo
- Cost of generating path on n nodes is of same order as for forward construction: $O(n)$

First application in financial/QMC context: Moskowitz & Caflisch (1996)

Brownian bridge construction

- Probabilistically equivalent to forward construction
- Influence of X_k on overall behavior of B is decreasing with k
  - stratified sampling
  - quasi-Monte Carlo
- Cost of generating path on n nodes is of same order as for forward construction: $O(n)$

First application in financial/QMC context: Moskowitz & Caflisch (1996)

Brownian bridge construction

- Probabilistically equivalent to forward construction
- Influence of X_k on overall behavior of B is decreasing with k
  - stratified sampling
  - quasi-Monte Carlo
- Cost of generating path on n nodes is of same order as for forward construction: $O(n)$

First application in financial/QMC context: Moskowitz & Caflisch (1996)

Unifying principle I: Linearity

Both constructions are of the form

$$B = AX$$

where

- $B = (B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})$
- $(X_1, \ldots, X_n)$ indep. std. normal variables
- A an $n \times n$ matrix

Unifying principle I: Linearity

Both constructions are of the form

$$B = AX$$

where

- $B = (B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})$
- $(X_1, \ldots, X_n)$ indep. std. normal variables
- A an $n \times n$ matrix

Unifying principle I: Linearity

Both constructions are of the form

$$B = AX$$

where

- $B = (B_{\frac{1}{n}}, \ldots, B_{\frac{n}{n}})$
- $(X_1, \ldots, X_n)$ indep. std. normal variables
- A an $n \times n$ matrix

Unifying principle I: Linearity

Both constructions are of the form

$$B = AX$$

where

- $B = (B_{\frac{1}{n}}, \dots, B_{\frac{n}{n}})$
- $(X_1, \dots, X_n)$ indep. std. normal variables
- A an $n \times n$ matrix

Unifying principle I: Linearity

Necessary and sufficient for B being a (discrete) Brownian path:

$$AA^\top = \frac{1}{n}\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 2 & \dots & 2 \\ 1 & 2 & 3 & \dots & 3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 3 & \dots & n \end{pmatrix} =: \Sigma$$

Unifying principle I: Linearity

Necessary and sufficient for B being a (discrete) Brownian path:

$$AA^\top = \frac{1}{n}\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 2 & \dots & 2 \\ 1 & 2 & 3 & \dots & 3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 3 & \dots & n \end{pmatrix} =: \Sigma$$

Unifying principle I: Linearity

For example:

- For the forward method

$$A = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix} =: S$$

  $SS^\top = \Sigma \dots$ Cholesky decomposition of Σ

- For the Brownian bridge construction

$$A = \text{something else}$$

Unifying principle I: Linearity

For example:

- For the forward method

$$A = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix} =: S$$

$SS^\top = \Sigma \dots$ Cholesky decomposition of Σ

- For the Brownian bridge construction

$$A = \text{something else}$$

Unifying principle I: Linearity

For example:

- For the forward method

$$A = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix} =: S$$

  $SS^\top = \Sigma \dots$ Cholesky decomposition of Σ

- For the Brownian bridge construction

$$A = \text{something else}$$

Unifying principle I: Linearity

For example:

- For the forward method

$$A = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix} =: S$$

  $SS^\top = \Sigma \dots$ Cholesky decomposition of Σ

- For the Brownian bridge construction

$$A = \text{something else}$$

Unifying principle I: Linearity

For example:

- For the forward method

$$A = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix} =: S$$

  $SS^\top = \Sigma \ldots$ Cholesky decomposition of Σ

- For the Brownian bridge construction

$$A = \text{something else}$$

Principal component analysis

a.k.a. PCA construction, Singular value construction

Take

$$A = VD^{1/2}$$

where $VDV^\top = \Sigma$ is the singular value decomposition of Σ.

- First component captures maximal variance
- second component captures maximal remaining variance
- and so on

First application in financial context: Acworth, Broadie & Glasserman (1996)

Principal component analysis

a.k.a. PCA construction, Singular value construction

Take

$$A = VD^{1/2}$$

where $VDV^\top = \Sigma$ is the singular value decomposition of Σ.

- First component captures maximal variance
- second component captures maximal remaining variance
- and so on

First application in financial context: Acworth, Broadie & Glasserman (1996)

Principal component analysis

a.k.a. PCA construction, Singular value construction

Take

$$A = VD^{1/2}$$

where $VDV^\top = \Sigma$ is the singular value decomposition of Σ.

- First component captures maximal variance
- second component captures maximal remaining variance
- and so on

First application in financial context: Acworth, Broadie & Glasserman (1996)

Principal component analysis

a.k.a. PCA construction, Singular value construction

Take

$$A = VD^{1/2}$$

where $VDV^\top = \Sigma$ is the singular value decomposition of Σ.

- First component captures maximal variance
- second component captures maximal remaining variance
- and so on

First application in financial context: Acworth, Broadie & Glasserman (1996)

Principal component analysis

a.k.a. PCA construction, Singular value construction

Take

$$A = VD^{1/2}$$

where $VDV^{\top} = \Sigma$ is the singular value decomposition of Σ.

- First component captures maximal variance
- second component captures maximal remaining variance
- and so on

First application in financial context: Acworth, Broadie & Glasserman (1996)

Principal component analysis

Disadvantage of PCA:

Multiplication with V costs $O(n^2)$

Or does it?

Scheicher (2007): PCA generation costs $O(n\log(n))$ by using the fast sine transform in dimension $2n+1$

Principal component analysis

Disadvantage of PCA:

Multiplication with V costs $O(n^2)$

Or does it?

Scheicher (2007): PCA generation costs $O(n\log(n))$ by using the fast sine transform in dimension $2n+1$

Principal component analysis

Disadvantage of PCA:

Multiplication with V costs $O(n^2)$

Or does it?

Scheicher (2007): PCA generation costs $O(n\log(n))$ by using the fast sine transform in dimension $2n+1$

Dependence on payoff

(Some people are never satisfied)

Papageorgiou (2002), Sloan & Wang (2011)

- Integration error depends on payoff
- Let f be a payoff function, A_1, A_2 two linear generation methods such that $E(f(A_1X))$ admits good/bad convergence under some QMC rule. Then there is a payoff g such that $E(g(A_2X))$ admits the same convergence under the same rule: $g(Y) = f(A_1A_2^{-1}Y)$.
- So there is no best generation method
- Suggests to look for optimal A (in some sense) for given payoff

Dependence on payoff

(Some people are never satisfied)

Papageorgiou (2002), Sloan & Wang (2011)

- Integration error depends on payoff
- Let f be a payoff function, A_1, A_2 two linear generation methods such that $E(f(A_1X))$ admits good/bad convergence under some QMC rule. Then there is a payoff g such that $E(g(A_2X))$ admits the same convergence under the same rule: $g(Y) = f(A_1A_2^{-1}Y)$.
- So there is no best generation method
- Suggests to look for optimal A (in some sense) for given payoff

Dependence on payoff

(Some people are never satisfied)

Papageorgiou (2002), Sloan & Wang (2011)

- Integration error depends on payoff
- Let f be a payoff function, A_1, A_2 two linear generation methods such that $E(f(A_1X))$ admits good/bad convergence under some QMC rule. Then there is a payoff g such that $E(g(A_2X))$ admits the same convergence under the same rule: $g(Y) = f(A_1A_2^{-1}Y)$.
- So there is no best generation method
- Suggests to look for optimal A (in some sense) for given payoff

Dependence on payoff

(Some people are never satisfied)

Papageorgiou (2002), Sloan & Wang (2011)

- Integration error depends on payoff
- Let f be a payoff function, A_1, A_2 two linear generation methods such that $E(f(A_1X))$ admits good/bad convergence under some QMC rule. Then there is a payoff g such that $E(g(A_2X))$ admits the same convergence under the same rule: $g(Y) = f(A_1A_2^{-1}Y)$.
- So there is no best generation method
- Suggests to look for optimal A (in some sense) for given payoff

Dependence on payoff

(Some people are never satisfied)

Papageorgiou (2002), Sloan & Wang (2011)

- Integration error depends on payoff
- Let f be a payoff function, A_1, A_2 two linear generation methods such that $E(f(A_1X))$ admits good/bad convergence under some QMC rule. Then there is a payoff g such that $E(g(A_2X))$ admits the same convergence under the same rule: $g(Y) = f(A_1A_2^{-1}Y)$.
- So there is no best generation method
- Suggests to look for optimal A (in some sense) for given payoff

Dependence on payoff

(Some people are never satisfied)

Papageorgiou (2002), Sloan & Wang (2011)

- Integration error depends on payoff
- Let f be a payoff function, A_1, A_2 two linear generation methods such that $E(f(A_1 X))$ admits good/bad convergence under some QMC rule. Then there is a payoff g such that $E(g(A_2 X))$ admits the same convergence under the same rule: $g(Y) = f(A_1 A_2^{-1} Y)$.
- So there is no best generation method
- Suggests to look for optimal A (in some sense) for given payoff

Unifying principle II: Orthogonality

In fact we can write any matrix A with $AA^\top = \Sigma$ as

$$A = SU$$

where

- U is an orthogonal matrix and
- S is the scaled summation defined earlier
- suggests we look for good/optimal U for our payoff
- PCA/BB provide good U for Asian options (and many other types)

Unifying principle II: Orthogonality

In fact we can write any matrix A with $AA^\top = \Sigma$ as

$$A = SU$$

where

- U is an orthogonal matrix and
- S is the scaled summation defined earlier
- suggests we look for good/optimal U for our payoff
- PCA/BB provide good U for Asian options (and many other types)

Unifying principle II: Orthogonality

In fact we can write any matrix A with $AA^\top = \Sigma$ as

$$A = SU$$

where

- U is an orthogonal matrix and
- S is the scaled summation defined earlier
- suggests we look for good/optimal U for our payoff
- PCA/BB provide good U for Asian options (and many other types)

Unifying principle II: Orthogonality

In fact we can write any matrix A with $AA^\top = \Sigma$ as

$$A = SU$$

where

- U is an orthogonal matrix and
- S is the scaled summation defined earlier
- suggests we look for good/optimal U for our payoff
- PCA/BB provide good U for Asian options (and many other types)

Unifying principle II: Orthogonality

In fact we can write any matrix A with $AA^\top = \Sigma$ as

$$A = SU$$

where

- U is an orthogonal matrix and
- S is the scaled summation defined earlier
- suggests we look for good/optimal U for our payoff
- PCA/BB provide good U for Asian options (and many other types)

Unifying principle II: Orthogonality

Makes method generic: every (quasi-)MC problem can by written as

$$E(f(X))$$

where X is a standard normal vector.

- $E(f(X)) = E(f(UX))$ for any orthogonal matrix U.
- Choose U such that variance/variation is concentrated on few variables

Sample application: BB/PCA-type generation for Lévy paths (L (2006))

Unifying principle II: Orthogonality

Makes method generic: every (quasi-)MC problem can by written as

$$E(f(X))$$

where X is a standard normal vector.

- $E(f(X)) = E(f(UX))$ for any orthogonal matrix U.
- Choose U such that variance/variation is concentrated on few variables

Sample application: BB/PCA-type generation for Lévy paths (L (2006))

Unifying principle II: Orthogonality

Makes method generic: every (quasi-)MC problem can by written as

$$E(f(X))$$

where X is a standard normal vector.

- $E(f(X)) = E(f(UX))$ for any orthogonal matrix U.
- Choose U such that variance/variation is concentrated on few variables

Sample application: BB/PCA-type generation for Lévy paths (L (2006))

Unifying principle II: Orthogonality

Makes method generic: every (quasi-)MC problem can by written as

$$E(f(X))$$

where X is a standard normal vector.

- $E(f(X)) = E(f(UX))$ for any orthogonal matrix U.
- Choose U such that variance/variation is concentrated on few variables

Sample application: BB/PCA-type generation for Lévy paths (L (2006))

Unifying principle II: Orthogonality
Examples

$$\Sigma = AA^\top \text{ with } A = SU$$

- Forward method: $U = \mathrm{Id}_{\mathbb{R}^n}$.
- Brownian Bridge: $U = H^{-1}$ where H is the Haar transform
- PCA: $U = S^{-1}VD^{1/2}$ (trivially)

Unifying principle II: Orthogonality

Examples

$$\Sigma = AA^\top \text{ with } A = SU$$

- Forward method: $U = \mathrm{Id}_{\mathbb{R}^n}$.
- Brownian Bridge: $U = H^{-1}$ where H is the Haar transform
- PCA: $U = S^{-1}VD^{1/2}$ (trivially)

Unifying principle II: Orthogonality

Examples

$$\Sigma = AA^\top \text{ with } A = SU$$

- Forward method: $U = \mathrm{Id}_{\mathbb{R}^n}$.
- Brownian Bridge: $U = H^{-1}$ where H is the Haar transform
- PCA: $U = S^{-1}VD^{1/2}$ (trivially)

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method (L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method (L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method (L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method

(L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method

(L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method (L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

One advantage of "orthogonal" formulation: there are many fast generation methods for BM besides forward, BB and PCA method (L 2011) reviews transforms that need at most $O(n \log n)$ operations

- Discrete Sine/Cosine transform
- Hartley-, Hilbert-, W- transform
- Walsh transform
- Haar transform, general wavelet transforms
- Tensor products of orthogonal transforms
- and more

Unifying principle II: Orthogonality

Theorem (Sloan & Wang (2011))

For every orthogonal transform there is a (theoretical) financial derivative for which the orthogonal transform is optimal, i.e. where variability is reduced to one dimension.

Empirical finding: for typical real world derivatives in the 1-dimensional BS-model, BB/PCA are good enough

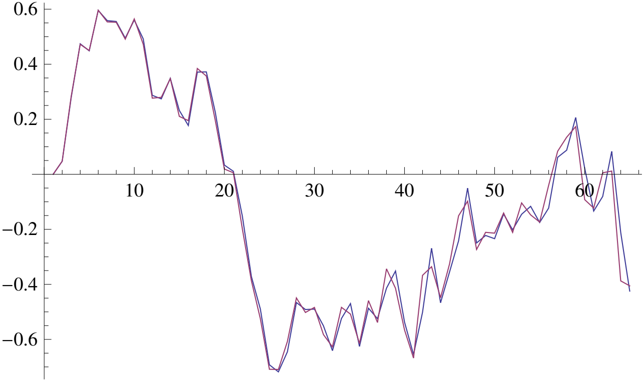
Unifying principle II: Orthogonality

Theorem (Sloan & Wang (2011))

For every orthogonal transform there is a (theoretical) financial derivative for which the orthogonal transform is optimal, i.e. where variability is reduced to one dimension.

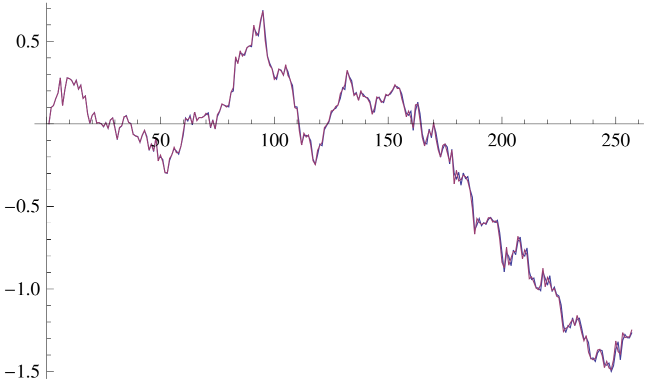
Empirical finding: for typical real world derivatives in the 1-dimensional BS-model, BB/PCA are good enough

Unifying principle II: Orthogonality



Comparison PCA/DCT-IV, $n = 64$

Unifying principle II: Orthogonality



Comparison PCA/DCT-IV, $n = 256$

Unifying principle II: Orthogonality
Numerical example

Derived from Sloan & Wang 2011: weighted Asian option.

$$
\begin{aligned}
f(B) &:= \max\left(\sum_{k=1}^{n} w_k S_{k/n} - K, 0\right) \\
&= \max\left(\frac{1}{n}\sum_{k=1}^{n} w_k S_0 \exp\left(\sigma B_{k/n} + (r - \frac{\sigma^2}{2})k/n\right) - K, 0\right),
\end{aligned}
$$

Unifying principle II: Orthogonality
Numerical example

OTP ... "orthogonal tensor product"

$$U = R \otimes R \otimes \ldots \otimes R \quad (\log_2(n) \text{ times})$$

R is a two-dimensional rotation about a fixed angle ($\phi = 0.4$).

Now choose weights $w_1, \ldots, w_n$ to make U a (near) optimal orthogonal transform.

Unifying principle II: Orthogonality

Numerical example

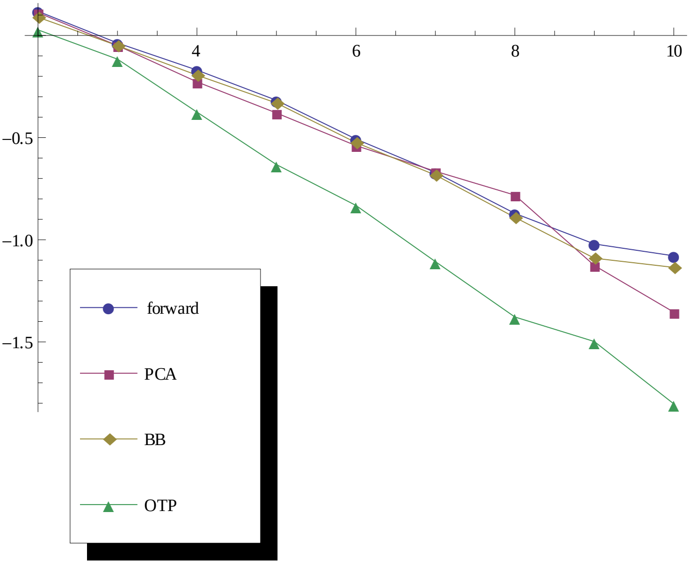
OTP ... "orthogonal tensor product"

$$U = R \otimes R \otimes \ldots \otimes R \quad (\log_2(n) \text{ times})$$

R is a two-dimensional rotation about a fixed angle ($\phi = 0.4$).
Now choose weights $w_1, \ldots, w_n$ to make U a (near) optimal orthogonal transform.

Unifying principle II: Orthogonality

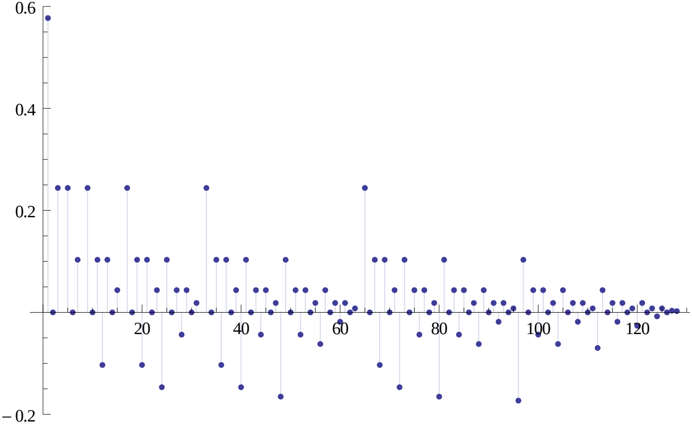
Numerical example



Unifying principle II: Orthogonality

Numerical example

The corresponding weights are rather exotic:



Unifying principle II: Orthogonality

At least on of these earlier constructions is actually useful in practice

Recall: Fast computation of path using PCA utilizes fast sine transform in dimension $2n + 1$

This is usually quite a bit slower than, e.g., the discrete cosine transform in dimension n, especially if $n = 2^k$

Unifying principle II: Orthogonality

At least on of these earlier constructions is actually useful in practice

Recall: Fast computation of path using PCA utilizes fast sine transform in dimension $2n+1$

This is usually quite a bit slower than, e.g., the discrete cosine transform in dimension n, especially if $n = 2^k$

Unifying principle II: Orthogonality

At least on of these earlier constructions is actually useful in practice

Recall: Fast computation of path using PCA utilizes fast sine transform in dimension $2n + 1$

This is usually quite a bit slower than, e.g., the discrete cosine transform in dimension n, especially if $n = 2^k$

Unifying principle II: Orthogonality

Theorem (L 2011)

Let

- $\Sigma = VDV^\top$ *be the PCA of* Σ
- C *the discrete cosine transform of type IV in dimension* n
- $d_n(P,Q)^2 := \sum_{l=1}^n \sum_{k=1}^n (P-Q)_{lk}^2$ *for* $n \times n$ *matrices* P, Q

Then for all $n \in \mathbb{N}$ *we have* $d_n(SC, VD^{\frac{1}{2}}) < 1$ *and*

$$\limsup_{n\to\infty} d_n(SC, VD^{\frac{1}{2}})^2 \le \frac{2\left(48-\pi^2\right)}{\left(\pi^2-24\right)^2} = 0.381\ldots .$$

Unifying principle II: Orthogonality

Theorem (L 2011)

Let

- $\Sigma = VDV^\top$ *be the PCA of* Σ
- C *the discrete cosine transform of type IV in dimension* n
- $d_n(P,Q)^2 := \sum_{l=1}^n \sum_{k=1}^n (P-Q)_{lk}^2$ *for* $n \times n$ *matrices* P, Q

Then for all $n \in \mathbb{N}$ *we have* $d_n(SC, VD^{\frac{1}{2}}) < 1$ *and*

$$\limsup_{n\to\infty} d_n(SC, VD^{\frac{1}{2}})^2 \leq \frac{2\left(48-\pi^2\right)}{\left(\pi^2-24\right)^2} = 0.381\ldots.$$

Unifying principle II: Orthogonality

Theorem (L 2011)

Let

- $\Sigma = VDV^\top$ *be the PCA of* Σ
- C *the discrete cosine transform of type IV in dimension* n
- $d_n(P,Q)^2 := \sum_{l=1}^n \sum_{k=1}^n (P-Q)_{lk}^2$ *for* $n \times n$ *matrices* P, Q

Then for all $n \in \mathbb{N}$ *we have* $d_n(SC, VD^{\frac{1}{2}}) < 1$ *and*

$$\limsup_{n\to\infty} d_n(SC, VD^{\frac{1}{2}})^2 \le \frac{2\left(48-\pi^2\right)}{\left(\pi^2-24\right)^2} = 0.381\ldots .$$

Unifying principle II: Orthogonality

Theorem (L 2011)

Let

- $\Sigma = VDV^\top$ *be the PCA of* Σ
- C *the discrete cosine transform of type IV in dimension* n
- $d_n(P,Q)^2 := \sum_{l=1}^n \sum_{k=1}^n (P-Q)_{lk}^2$ *for* $n \times n$ *matrices* P, Q

Then for all $n \in \mathbb{N}$ *we have* $d_n(SC, VD^{\frac{1}{2}}) < 1$ *and*

$$\limsup_{n\to\infty} d_n(SC, VD^{\frac{1}{2}})^2 \le \frac{2\left(48-\pi^2\right)}{\left(\pi^2-24\right)^2} = 0.381\ldots .$$

Unifying principle II: Orthogonality

Theorem (L 2011)

Let

- $\Sigma = VDV^\top$ *be the PCA of* Σ
- C *the discrete cosine transform of type IV in dimension* n
- $d_n(P,Q)^2 := \sum_{l=1}^n \sum_{k=1}^n (P-Q)_{lk}^2$ *for* $n \times n$ *matrices* P, Q

Then for all $n \in \mathbb{N}$ *we have* $d_n(SC, VD^{\frac{1}{2}}) < 1$ *and*

$$\limsup_{n\to\infty} d_n(SC, VD^{\frac{1}{2}})^2 \leq \frac{2\left(48-\pi^2\right)}{\left(\pi^2-24\right)^2} = 0.381\ldots .$$

Unifying principle II: Orthogonality

Thus we have found a construction method that

- gives essentially same result as PCA
- uses at most half the time for path generation

Unifying principle II: Orthogonality

Thus we have found a construction method that

- gives essentially same result as PCA
- uses at most half the time for path generation

Unifying principle II: Orthogonality

Thus we have found a construction method that

- gives essentially same result as PCA
- uses at most half the time for path generation

Optimal orthogonal transforms

Imai & Tan (2007): Find U that works best

Goal: Find good U that admits fast matrix-vector multiplication
(Irrgeher & L, ongoing research)

Useful only for derivatives or models that depend on several Brownian paths – or that are entirely different.

Optimal orthogonal transforms

Imai & Tan (2007): Find U that works best

Goal: Find good U that admits fast matrix-vector multiplication (Irrgeher & L, ongoing research)

Useful only for derivatives or models that depend on several Brownian paths – or that are entirely different.

Optimal orthogonal transforms

Imai & Tan (2007): Find U that works best

Goal: Find good U that admits fast matrix-vector multiplication (Irrgeher & L, ongoing research)

Useful only for derivatives or models that depend on several Brownian paths – or that are entirely different.

Thank you for your attention!