CFM-Imperial Distinguished Lecture Series

The Volatility Surface

Lecture 2: The SVI arbitrage-free volatility surface parameterization

Jim Gatheral
Department of Mathematics

Baruch COLLEGE
The City University of New York

Outline of Lecture 2

- No-arbitrage constraints on the tail behavior of implied volatility.
- The SVI parameterization of the volatility smile and its variants.
  - Sufficient conditions for no calendar-spread arbitrage.
  - Necessary and sufficient conditions for no calendar-spread arbitrage.
  - Arbitrage on a slice.
- Generating an implied volatility surface from raw index option data.
- Visualization:
  - Total variance plot.
  - 3D plot.
  - Local variance plot.
- The SVI square root parameterization.
- Fitting SVI subject to no-arbitrage constraints.

What is a volatility surface parameterization for?

- Options market making needs an easy to calibrate functional form.
- Computing the local volatility surface for risk management of portfolios of exotic options.
- Dimensional reduction for the analysis of volatility surface dynamics.

Roger Lee's Moment Formula

- [Roger Lee][6] shows that implied variance is bounded above by a function linear in the log-strike $k = \log(K/F)$ as $|k| \to \infty$.
  - The maximum slope of total implied variance $w(k,T) = \sigma_{BS}^2(k,T)\,T$ is 2.
- He shows how to relate the gradients of the wings of the upper bound of the implied variance skew to the maximal finite moments of the underlying process.
- Lee's derivation assumes only the existence of a martingale measure: it makes no assumptions on the distribution of underlying returns. His result is completely model-independent.

> **Roger Lee's Lemma 3.1**
>
> There exists $k^\star > 0$ such that for all $k > k^\star$,
>
> $$\sigma_{BS}^2(k) < 2\,\frac{|k|}{T}.$$

Proof.

We only need to show that $ C_{BS}(k,*{BS}(k),) < C{BS}(k,) $ whenever $k > k^\star$.
On the LHS, we have

$$\lim_{k\to\infty} C_{BS}\left(k, \sigma_{BS}(k)\sqrt{T}\right) = 0$$

and on the RHS, we have

The left wing

Specifically, let $q^* := \sup\left\{q : \mathbb{E}\, S_T^{-q} < \infty\right\}$ and

$$\beta^* := \limsup_{k\to-\infty} \frac{\sigma_{BS}^2(k,T)\,T}{|k|}$$

Then $\beta^* \in [0,2]$,

$$q^* = \frac{1}{2}\left(\frac{1}{\sqrt{\beta^*}} - \frac{\sqrt{\beta^*}}{2}\right)^2$$

and inverting this, we obtain $\beta^* = g(q^*)$ with

$$g(x) = 2 - 4\left[\sqrt{x^2+x} - x\right]$$

The right wing

Similarly, let $p^* := \sup\left\{p : \mathbb{E}\, S_T^{1+p} < \infty\right\}$ and

$$\alpha^* := \limsup_{k\to+\infty} \frac{\sigma_{BS}^2(k,T)\,T}{|k|}$$

Then $\alpha^* \in [0,2]$,

$$p^* = \frac{1}{2}\left(\frac{1}{\sqrt{\alpha^*}} - \frac{\sqrt{\alpha^*}}{2}\right)^2$$

and as for the left wing, it follows that $\alpha^* = g(p^*)$.

Benaim and Friz

- With $F(x)$ the CDF, Benaim and Friz[1] go on to show that Roger Lee's upper bound ($\limsup$) may be replaced by a limit in most practical cases.
- Then we may write for the right tail

  (1)

$$\frac{\sigma_{BS}(k,T)^2\,T}{k} \sim g\left(-1 - \frac{\log\left[1-F(k)\right]}{k}\right) \text{ as } k\to\infty$$

  and for the left tail

  (2)

$$\frac{\sigma_{BS}(-k,T)^2\,T}{k} \sim g\left(\frac{-\log F(-k)}{k}\right) \text{ as } k\to\infty$$

- By substituting the tail-behavior of F into equations (1) and (2), we can deduce the full tail behavior of the smile, not just Roger Lee's upper bound.

Example: Black-Scholes

$$\begin{aligned} 1-F(k) = &\int_k^\infty \frac{1}{\sqrt{2\,\pi\,\sigma^2\,T}}\, e^{-y^2/(2\sigma^2\,T)}\, dy \\ \sim &\frac{1}{\sqrt{2\,\pi}}\,\frac{e^{-k^2/(2\,\sigma^2\,T)}}{k} \text{ as } k\to\infty \end{aligned}$$

Then

$$\log\left[1-F(k)\right] \sim -\frac{k^2}{2\,\sigma^2\,T} \text{ as } k\to\infty$$

and

$$\frac{\sigma_{BS}(k,T)^2\,T}{k} \sim g\left(-1 + \frac{k}{2\,\sigma^2\,T}\right) \sim \frac{2\,\sigma^2\,T}{2\,k} \text{ as } k\to\infty$$

It follows that

$$\sigma_{BS}(k,T)^2 \sim \sigma^2 \text{ as } k\to\infty$$

Implications of the moment formula

- Implied variance is linear in k as $k\to\infty$ for stochastic volatility models.
- So, if we want a parametrization of the implied variance surface consistent with stochastic volatility, it needs to be linear in the wings!
  - and it needs to be curved in the middle - many conventional parameterizations of the volatility surface are quadratic for example.

History of SVI

- SVI was originally devised at Merrill Lynch in 1999 and subsequently written up in detail in [Gatheral and Jacquier][5].
- SVI has two key properties that have led to its subsequent popularity with practitioners:
  - For a fixed time to expiry t, the implied Black-Scholes variance $\sigma^2_{\mathrm{BS}}(k,t)$ is linear in the log-strike k as $|k| \to \infty$ consistent with Roger Lee's moment formula [Roger Lee][6].
  - It is relatively easy to fit listed option prices whilst ensuring no calendar spread arbitrage.
- As shown in [Gatheral and Jacquier][4], the SVI parameterization is not arbitrary in the sense that the large-maturity limit of the Heston implied volatility smile is exactly SVI.

Characterisation of static arbitrage

Definition 2.1

A volatility surface is free of static arbitrage if and only if the following conditions are satisfied:

i. it is free of calendar spread arbitrage;

ii. each time slice is free of butterfly arbitrage.

Calendar spread arbitrage

Lemma 2.2

If dividends are proportional to the stock price, the volatility surface w is free of calendar spread arbitrage if and only if

$$\partial_t w(k,t) \geq 0, \quad \text{for all } k \in \mathbb{R} \text{ and } t > 0.$$

- Thus there is no calendar spread arbitrage if there are no crossed lines on a total variance plot.

Proof of Lemma 2.2

- For any martingale X_t and $t_2 t_1$,
$$\mathbb{E}\left[(X_{t_2} - L)^+\right] \geq \mathbb{E}\left[(X_{t_1} - L)^+\right]$$
- Define $F_{0,t} := \mathbb{E}[S_t | \mathcal{F}_0]$.
- Let C_i be options with strikes K_i and expirations t_i with $t_2 > t_1$. Suppose the two options have the same moneyness so that
$$\frac{K_1}{F_{0,t_1}} = \frac{K_2}{F_{0,t_2}} =: e^k$$
- Then, if dividends are proportional, $X_t := S_t / F_{0,t}$ is a martingale and
$$\frac{C_2}{K_2} = e^{-k}\mathbb{E}\left[\left(X_{t_2} - e^k\right)^+\right] \geq e^{-k}\mathbb{E}\left[\left(X_{t_1} - e^k\right)^+\right] = \frac{C_1}{K_1}.$$
- So, if dividends are proportional, keeping moneyness constant, option prices are non-decreasing in time to expiration.
- Let $w_t := \sigma_{BS}^2(k,T)\,T$.
- The Black-Scholes formula for the non-discounted value of an option may be expressed in the form $C_{BS}(k, w_t)$ with $C_{BS}(\cdot)$ strictly increasing in the total implied variance w_t.
- It follows that for fixed k, we must have the total implied variance w_t non-decreasing with respect to time to expiration:

No calendar spread arbitrage condition

$$\partial_t w(k;\chi_t) \geqslant 0$$

where χ_t denotes the t-dependent parameters of our fit.

Butterfly arbitrage

Definition 2.3

A slice is said to be free of butterfly arbitrage if the corresponding density is non-negative.

Now introduce the function $g : \mathbb{R} \to \mathbb{R}$ defined by

$$g(k) := \left(1 - \frac{k w'(k)}{2w(k)}\right)^2 - \frac{w'(k)^2}{4}\left(\frac{1}{w(k)} + \frac{1}{4}\right) + \frac{w''(k)}{2}.$$

Lemma 2.4

A slice is free of butterfly arbitrage if and only if $g(k) \geq 0$ for all $k \in \mathbb{R}$ and $\lim_{k \to +\infty} d_+(k) = -\infty$.

The raw SVI parameterization

For a given parameter set $\chi_R = \{a, b, \rho, m, \sigma\}$, the *raw SVI parameterization* of total implied variance reads:

Raw SVI parameterization

$$w(k;\chi_R) = a + b\left\{\rho\,(k-m) + \sqrt{(k-m)^2 + \sigma^2}\right\}$$

where $a \in \mathbb{R}$, $b \geq 0$, $|\rho| < 1$, $m \in \mathbb{R}$, $\sigma > 0$, and the obvious condition $a + b\,\sigma\sqrt{1-\rho^2} \geq 0$, which ensures that $w(k,\chi_R) \geq 0$ for all $k \in \mathbb{R}$. This condition ensures that t minimum of the function $w(\cdot,\chi_R)$ is non-negative.

Set up environment

In [1]:
```
load_ext rpy2.ipython
```

In [2]:
```
%%R

download.file(url="http://mfe.baruch.cuny.edu/wp-content/uploads/2015/03/ImperialLecture2.zip", destfile="ImperialLecture2.zip
)
unzip(zipfile="ImperialLecture2.zip")

library(stinepack)
```

```
trying URL 'http://mfe.baruch.cuny.edu/wp-content/uploads/2015/03/ImperialLecture2.zip'
Content type 'application/zip' length 193359 bytes (188 Kb)
opened URL
==================================================
downloaded 188 Kb
```

In [3]:
```
%%R

setwd("./ImperialLecture2")

source("BlackScholes.R")
source("plotSVI.R")
source("sviJW.R")
source("plotIvols.R")
source("sviarbitrage.R")
source("optionMetricsToIvols.R")
source("sviRoots.R")
source("sviFit0.R")
source("sviSqrtFit.R")
source("computeImpliedVols.R")
source("sviFitQuarticRoots.R")
source("sviVolSurface.R")
source("svi.R")
```

An SVI example

Here's a specific choice of parameters:

In [4]:
```
%%R

sviparams <- list(a = 0.04, b = 0.4, sig = 0.1, rho = - 0.4, m = 0.1);
```

And here's how the SVI smile looks with these parameters:

In [5]:
```
%%R
curve(svi(sviparams,x),from=-1,to=1,col="red",xlab="Log-strike k",ylab=expression(paste("Implied variance  ",sigma^2*T)));
```

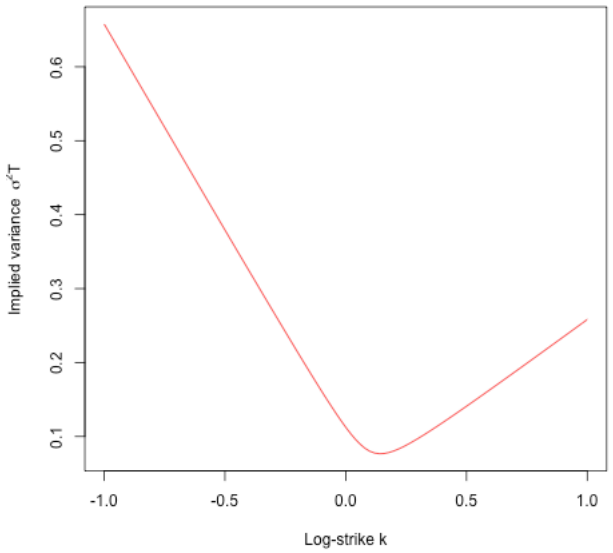


Figure 1: The SVI smile with $a = 0.04$, $b = 0.4$, $\sigma = 0.1$, $\rho = -0.4$, $m = 0$

Meaning of raw SVI parameters

Changes in the parameters have the following effects:

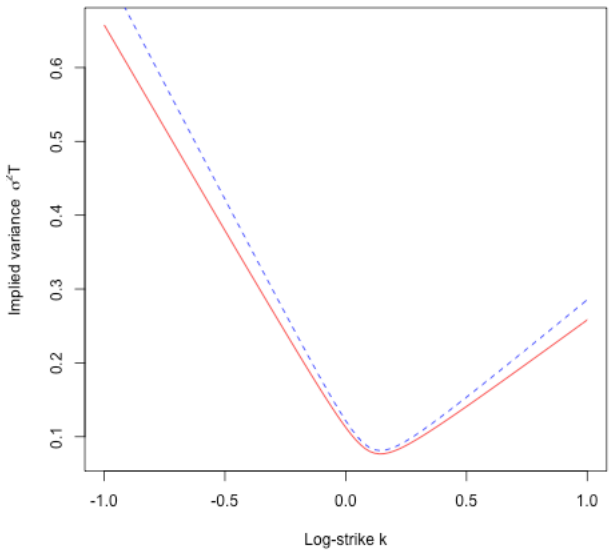
- Increasing a increases the general level of variance, a vertical translation of the smile;
- Increasing b increases the slopes of both the put and call wings, tightening the smile;
- Increasing ρ decreases (increases) the slope of the left(right) wing, a counter-clockwise rotation of the smile;
- Increasing m translates the smile to the right;
- Increasing σ reduces the at-the-money (ATM) curvature of the smile.

We can convince ourselves that if the wings are linear in k, we need 5 and only 5 parameters to cover all reasonable transformations of the volatility smile.

Demonstrate the effect of changing raw SVI parameters:

In [6]:
```
%%R

sviparams <- list(a = 0.04, b = 0.4, sig = 0.1, rho = - 0.4, m = 0.1)
curve(svi(sviparams,x),from=-1,to=1,col="red",xlab="Log-strike k",ylab=expression(paste("Implied variance  ",sigma^2*T)))
sviparams2 <- sviparams
sviparams2$b <- sviparams$b + 0.05
curve(svi(sviparams2,x),from=-1,to=1,col="blue",add=T, lty=2)
```



The natural SVI parameterization

For a given parameter set $\chi_N = \{\Delta, \mu, \rho, \omega, \zeta\}$, the *natural SVI parameterization* of total implied variance reads:

Natural SVI parameterization

$$w(k;\chi_N) = \Delta + \frac{\omega}{2}\left\{1 + \zeta\rho\,(k-\mu) + \sqrt{(\zeta(k-\mu)+\rho)^2 + \left(1-\rho^2\right)}\right\}$$

where $\omega \geq 0$, $\Delta \in \mathbb{R}$, $\mu \in \mathbb{R}$, $|\rho| < 1$ and $\zeta > 0$.

- This parameterization is a natural generalization of the time ∞ Heston smile explored in [Gatheral and Jacquier][4].

The SVI Jump-Wings (SVI-JW) parameterization

- Neither the raw SVI nor the natural SVI parameterizations are intuitive to traders.
- There is no reason to expect these parameters to be particularly stable.
- The *SVI-Jump-Wings (SVI-JW) parameterization* of the implied variance v (rather than the implied total variance w) was inspired by a similar parameterization attributed to Tim Klassen, then at Goldman Sachs.

SVI-JW

For a given time to expiry $t > 0$, and an SVI total variance smile $w(k)$, the SVI-JW parameters are defined as follows:

SVI-JW parameterization

Interpretation of SVI-JW parameters

The SVI-JW parameters have the following interpretations:

- v_t gives the ATM variance;
- ψ_t gives the ATM skew;
- p_t gives the slope of the left (put) wing;
- c_t gives the slope of the right (call) wing;
- $\tilde{v}_t$ is the minimum implied variance.

Demonstrate the effect of changing SVI-JW parameters:

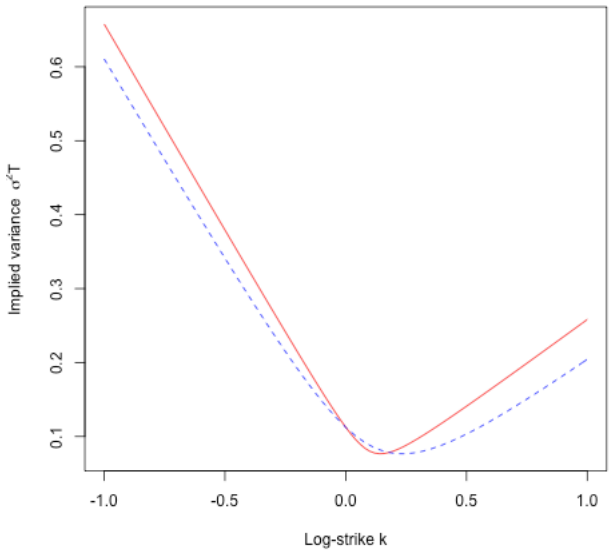
In [7]:
```
%%R

(sviJWparams <- sviToJw(sviparams,1))
```

```
         vt       psit       pt        ct    varmint texp
1 0.1125685 -0.6599499 1.669089 0.7153239 0.07666061    1
```

In [8]:
```
%%R

curve(svi(sviparams,x),from=-1,to=1,col="red",xlab="Log-strike k",ylab=expression(paste("Implied variance  ",sigma^2*T)))
sviJWparams2 <- sviJWparams
sviJWparams2$psit <- sviJWparams$psit + .2
sviparams2 <- jwToSvi(sviJWparams2)
curve(svi(sviparams2,x),from=-1,to=1,col="blue",add=T, lty=2)
```



Scaling of SVI Jump-Wings parameters with volatility

Note that, as defined here,

$$\psi_t = \left.\frac{\partial \sigma_{BS}(k)}{\partial k}\right|_{k=0}$$

The choice of volatility skew as the skew measure rather than variance skew for example, reflects the empirical observation that volatility is roughly lognormally distributed. Specifica we show in Chapter 7 of *The Volatility Surface* that if the SDE for variance is of the form:

$$dv = -\lambda\,(v - \bar{v})\,dt + \eta\,\sqrt{v}\,\beta(v)\,dZ$$

we should have

$$\frac{\partial}{\partial k}\,\sigma_{BS}(k,T)^2 \;\approx\; \frac{\rho\,\eta\,\beta(v)}{\lambda'\,T}\left\{1 - \frac{\left(1 - e^{-\lambda' T}\right)}{\lambda'\,T}\right\} \propto \beta(v)$$

with $' = -, , , (v)$.

Thus

$$\left.\frac{\partial \sigma_{BS}(k)}{\partial k}\right|_{k=0} \approx \text{const.}$$

independent of volatility implies that

$$\beta(v) \sim \sqrt{v}$$

and therefore that the variance (volatility) process is lognormal.

This consistency of the SVI-JW parameterization with empirical volatility dynamics leads to greater parameter stability over time.

Scaling of SVI Jump-Wings parameters with time to expiration

- If smiles scaled perfectly as $1/\sqrt{w_t}$ (effectively $1/\sqrt{t}$ in practice), SVI-JW parameters would be constant, independent of the slice t.
  - This makes it easy to extrapolate the SVI surface to expirations beyond the longest expiration in the data set.
- Since both scaling features are roughly consistent with empirical observation, we expect (and see) greater parameter stability over time.
  - Traders can keep parameters in their heads.

SVI slices may cross at no more than four points

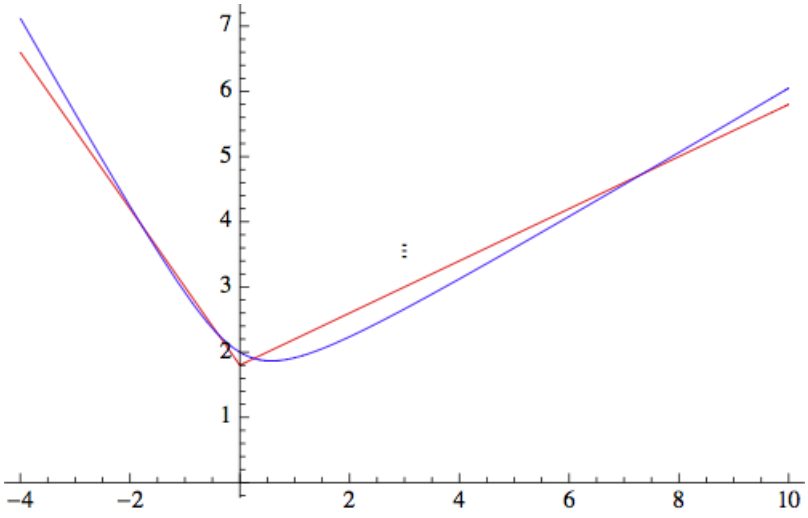


Figure 2: Lines cross at four points here.

Condition for no calendar spread arbitrage

Lemma 3.1

Two raw SVI slices admit no calendar spread arbitrage if a certain quartic polynomial has no real root.

Ferrari Cardano

The idea is as follows:

- Two total variance slices cross if
- Rearranging and squaring gives a quartic polynomial equation of the form

$$\alpha_4\, k^4 + \alpha_3\, k^3 + \alpha_2\, k^2 + \alpha_1\, k + \alpha_0 = 0,$$

where each of the coefficients are lengthy yet explicit expressions in terms of the raw SVI parameters.

- If this quartic polynomial has no real root, then the slices do not intersect.
- Roots of a quartic polynomial are known in closed-form thanks to [Cardano][2].

SVI butterfly arbitrage

Recall the definition:

$$g(k) := \left(1 - \frac{kw'(k)}{2w(k)}\right)^2 - \frac{w'(k)^2}{4}\left(\frac{1}{w(k)} + \frac{1}{4}\right) + \frac{w''(k)}{2}.$$

- The highly nonlinear behavior of g makes it seemingly impossible to find general conditions on the parameters that would eliminate butterfly arbitrage.
- We now provide an example where the no-butterfly-arbitrage condition is violated.

In [9]:
```
%%R
# Butterfly g(.) function
g <- function(sviparams,k){

  a <- sviparams$a;
  b <- sviparams$b;
  sig <- sviparams$sig;
  rho <- sviparams$rho;
  m <- sviparams$m;

  discr <- sqrt((k-m)*(k-m) + sig*sig);
  w <- a + b *(rho*(k-m)+ discr);
  dw <- b*rho + b *(k-m)/discr;
  d2w <- b*sig^2/(discr*discr*discr);

  return(1 - k*dw/w + dw*dw/4*(-1/w+k*k/(w*w)-4) +d2w/2);
}
```

Axel Vogt post on Wilmott.com

AVt
Senior Member

Posts: 971
Joined: Dec 2001

Thu Apr 06, 06 08:37 PM

It works for observables and far beyond for extrapolation.

But for a (theoretical) experiment try the following data

a = -.40998372001772e-1,
b = .13308181151379,
m = .35858898335748,
rho = .30602086142471,
sigma = .41531878803777

The Vogt smile

In [10]:
```
%%R
#-----------------------------------------------------------------
# Axel Vogt butterfly arbitrage example

a <- -0.0410
b <- 0.1331
m <- 0.3586
rho <- 0.3060
sig <- 0.4153

vogtParams <- as.data.frame(list(a=a,b=b,m=m,rho=rho,sig=sig))
print(vogtJW <- sviToJw(vogtParams,1))  # SVI-JW parameters
```

```
          vt       psit        pt       ct   varmint texp
1 0.01742625 -0.1752111 0.6997381 1.316798 0.0116249    1
```

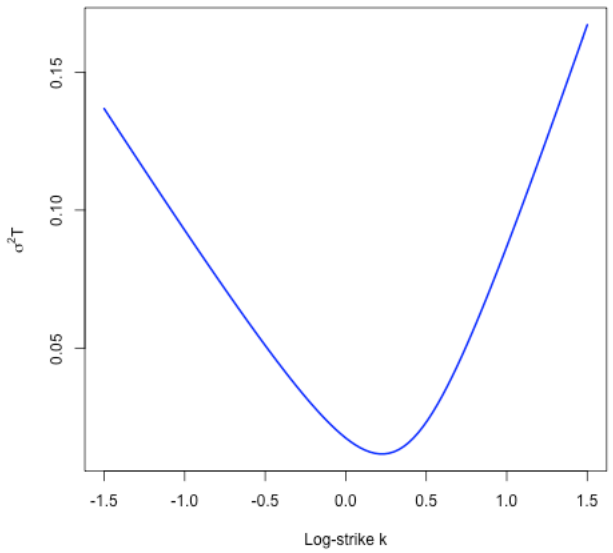
Now let's take a look at the smile and graph the function $g(k)$:

In [11]:
```
%%R



# Plot Vogt smile (smileSVI)
r <- 1.5
curve(svi(vogtParams,x),from=-r,to=r,col="blue",lwd=2,
      ylab=expression(w=sigma^2*T),xlab="Log-strike k")


# Plot graphs of g (ggraphSVI)
curve(g(vogtParams,x),from=-r,to=r,col="blue",n=500,lwd=2,ylab=expression(g(k)),xlab="Log-strike k")
lines(c(-r,r),c(0,0),lty=2)
```



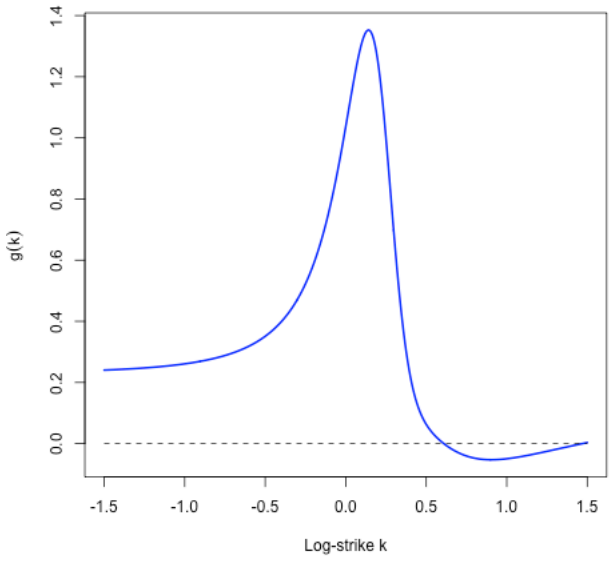


Figure 3: Plots of the total variance smile w and the function g, using Axel Vogt's parameters

- We see that the function $g(\cdot)$ and so the density goes negative with the Vogt parameters.

We knew already that there could be arbitrage on a slice with SVI and Axel Vogt has given us an embarrassing example. In fact, we can generate arbitrage even with $\rho = 0$. Let's re the plots:

In [12]:
```
%%R

vogtParams0 <- as.data.frame(list(a=a,b=b,m=m,rho=0,sig=sig))

# Plot Vogt smile (smileSVI)
r <- 1.5
curve(svi(vogtParams0,x),from=-r,to=r,col="blue",lwd=2,
      ylab=expression(w=sigma^2*T),xlab="Log-strike k")


# Plot graphs of g (ggraphSVI)
curve(g(vogtParams0,x),from=-r,to=r,col="blue",n=500,lwd=2,ylab=expression(g(k)),xlab="Log-strike k")
lines(c(-r,r),c(0,0),lty=2)
```

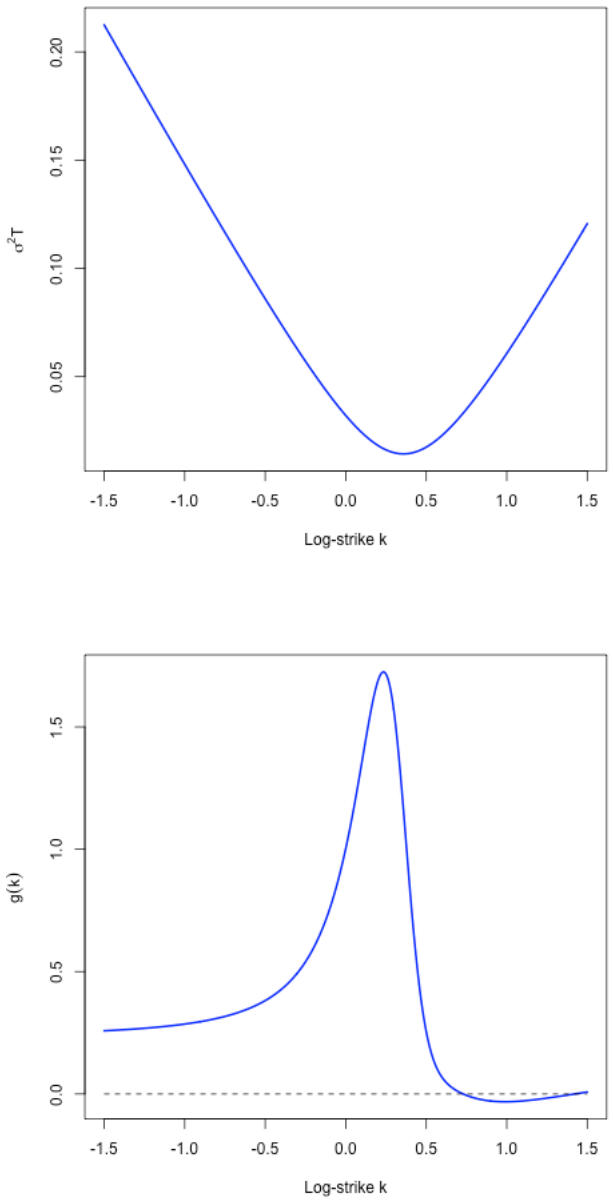


Figure 3: Plots of the total variance smile $w(k,T)$ (top) and the function g(k) (bottom), using Axel Vogt's parameters.

Until 2012 or so, I thought there was no simple way to guarantee the absence of butterfly arbitrage.

Surface SVI

Consider now the following extension of the natural SVI parameterization:

Surface SVI (SSVI) parameterization

(3)
$$w(k,\theta_t) = \frac{\theta_t}{2}\left\{1 + \rho\varphi(\theta_t)k + \sqrt{(\varphi(\theta_t)k+\rho)^2 + (1-\rho^2)}\right\}$$

with $\theta_t > 0$ for $t > 0$, and where φ is a smooth function from $(0,\infty)$ to $(0,\infty)$ such that the limit $\lim_{t\to 0}\theta_t\varphi(\theta_t)$ exists in $\mathbb{R}$.

Interpretation of SSVI

- This representation amounts to considering the volatility surface in terms of ATM variance time, instead of standard calendar time.
- The ATM total variance is $\theta_t = \sigma^2_{\mathrm{BS}}(0,t)\,t$ and the ATM volatility skew is given by

$$\partial_k \sigma_{\mathrm{BS}}(k,t)|_{k=0} = \frac{1}{2\sqrt{\theta_t t}}\partial_k w(k,\theta_t)\Big|_{k=0} = \frac{\rho\sqrt{\theta_t}}{2\sqrt{t}}\varphi(\theta_t).$$

- The smile is symmetric around at-the-money if and only if $\rho = 0$, a well-known property of stochastic volatility models.

Conditions on SSVI for no calendar spread arbitrage

Theorem 4.1

The SSVI surface (3) is free of calendar spread arbitrage if and only if

1. $\partial_t\theta_t \geq 0$, for all $t \geq 0$;
2. $0 \leq \partial_\theta(\theta\varphi(\theta)) \leq \frac{1}{\rho^2}\left(1+\sqrt{1-\rho^2}\right)\varphi(\theta)$, for all $\theta > 0$,

where the upper bound is infinite when $\rho = 0$.

- In particular, SSVI is free of calendar spread arbitrage if:
  - the skew in total variance terms is monotonically increasing in trading time and
  - the skew in implied variance terms is monotonically decreasing in trading time.
- In practice, any reasonable skew term structure that a trader defines will have these properties.

Conditions on SSVI for no butterfly arbitrage

Theorem 4.2

The volatility surface (3) is free of butterfly arbitrage if the following conditions are satisfied for all $\theta > 0$:

1. $\theta\varphi(\theta)\,(1 + |\rho|) < 4$;
2. $\theta\varphi(\theta)^2\,(1 + |\rho|) \leq 4$.

- Condition 1 needs to be a strict inequality so that $\lim_{k\to+\infty} d_+(k) = -\infty$ and the SVI density integrates to one.

Are these conditions necessary?

Lemma 4.2

The volatility surface (3) is free of butterfly arbitrage only if

$$\theta\varphi(\theta)\,(1 + |\rho|) \leq 4, \quad \text{for all } \theta > 0.$$

Moreover, if

$\theta\varphi(\theta)\,(1 + |\rho|) = 4$, the surface (3) is free of butterfly arbitrage only if

$$\theta\varphi(\theta)^2\,(1 + |\rho|) \leq 4.$$

- So the theorem is almost if-and-only-if.

The Roger Lee arbitrage bounds

- The asymptotic behavior of the surface (3) as $|k|$ tends to infinity is

$$w(k, \theta_t) = \frac{(1 \pm \rho)\,\theta_t}{2}\varphi(\theta_t)\,|k| + \mathcal{O}(1), \quad \text{for any } t > 0.$$

- Thus the condition $\theta\varphi(\theta)\,(1 + |\rho|) \leq 4$ of Theorem 4.2 corresponds to the upper bound of 2 on the asymptotic slope established by Roger Lee.
  - Again, Condition 1 of the theorem is necessary.

No static arbitrage with SSVI

Corollary 4.1

The SSVI surface (3) is free of static arbitrage if the following conditions are satisfied:

1. $\partial_t \theta_t \geq 0$, for all $t > 0$
2. $0 \leq \partial_\theta(\theta\varphi(\theta)) \leq \frac{1}{\rho^2}\left(1 + \sqrt{1-\rho^2}\right)\varphi(\theta)$, for all $\theta > 0$;
3. $\theta\varphi(\theta)\,(1 + |\rho|) < 4$, for all $\theta > 0$;
4. $\theta\varphi(\theta)^2\,(1 + |\rho|) \leq 4$, for all $\theta > 0$.

- A large class of simple closed-form arbitrage-free volatility surfaces!

A Heston-like surface

Example 4.1

The function φ defined as

$$\varphi(\theta) = \frac{1}{\lambda\theta}\left\{1 - \frac{1 - \mathrm{e}^{-\lambda\theta}}{\lambda\theta}\right\},$$

with $\lambda \geq (1 + |\rho|)/4$ satisfies the conditions of Corollary 4.1.

- This function is consistent with the implied variance skew in the Heston model as shown in [The Volatility Surface][3] (equation 3.19).

A power-law surface

Example 4.2

The choice

$$\varphi(\theta) = \frac{\eta}{\theta^\gamma\,(1+\theta)^{1-\gamma}}$$

gives a surface that is completely free of static arbitrage provided that $\gamma \in (0, 1/2]$ and $\eta\,(1 + |\rho|) \leq 2$.

- This function is more consistent with the empirically-observed term structure of the volatility skew.

Even more flexibility...

Theorem 4.3

Let $(k, t) \mapsto w(k, t)$ be a SSVI volatility surface (3) satisfying the conditions of Corollary 4.1 (in particular free of static arbitrage), and $\alpha : \mathbb{R}_+ \to \mathbb{R}_+$ a non-negative and increasing function of time. Then the volatility surface $(k, t) \mapsto w_\alpha(k, \theta_t) := w(k, \theta_t) + \alpha_t$ is also free of static arbitrage.

- Corollary 4.1 gives us the freedom to match three features of one smile (level, skew, and curvature say) but only two features of all the other smiles (level and skew say), subject of course to the given smiles being themselves arbitrage-free.
- Theorem 4.3 may allow us to match an additional feature of each smile through α_t.

How to eliminate butterfly arbitrage

- We have shown how to define a volatility smile that is free of butterfly arbitrage.
- This smile is completely defined given three observables.
  - The ATM volatility and ATM skew are obvious choices for two of them.
  - The most obvious choice for the third observable in equity markets would be the asymptotic slope for k negative and in FX markets and interest rate markets, perhaps the ATM curvature of the smile might be more appropriate.

How to fix butterfly arbitrage

- Supposing we choose to fix the SVI-JW parameters v_t, ψ_t and p_t of a given SVI smile, we may guarantee a smile with no butterfly arbitrage by choosing the remaining parameters c_t' and $\widetilde{v}_t'$ according to SSVI as

$$c_t' = p_t + 2\,\psi_t, \quad \text{and} \quad \widetilde{v}_t' = v_t\,\frac{4 p_t c_t'}{\left(p_t + c_t'\right)^2}.$$

- That is, given a smile defined in terms of its SVI-JW parameters, we are guaranteed to be able to eliminate butterfly arbitrage by changing the call wing c_t and the minimum varian $\widetilde{v}_t$, both parameters that are hard to calibrate with available quotes in equity options markets.

Example: Fixing the Vogt smile

- The SVI-JW parameters corresponding to the Vogt smile are:

- We know then that choosing $\left(c_t, \widetilde{v}_t\right) = (0.3493158, 0.01548182)$ must give a smile free of butterfly arbitrage.
- There must exist some pair of parameters $\{c_t, \widetilde{v}_t\}$ with $c_t \in (0.349, 1.317)$ and $\widetilde{v}_t \in (0.0116, 0.0155)$ such that the new smile is free of butterfly arbitrage and is as close as possible to the original one in some sense.

The R-code is below:

In [13]:
```
%%R

t <- 1
w <- a+b*(-rho*m + sqrt(m^2+sig^2))
v <- w/t
psi <- 1/sqrt(w)*b/2*(-m/sqrt(m^2+sig^2)+rho)
p <- 1/sqrt(w)*b*(1-rho)
c <- 1/sqrt(w)*b*(1+rho)
wtilde <- (a+b*sig*sqrt(1-rho^2))
vtilde <- wtilde/t

cprime <- vogtJW$pt + 2*vogtJW$psit
vtildeprime <- vogtJW$vt*4*vogtJW$pt*cprime/(vogtJW$pt+cprime)^2;

vogtFixedJW <- vogtJW
vogtFixedJW$ct <- cprime
vogtFixedJW$varmint <- vtildeprime

print(as.data.frame(rbind(vogtJW,vogtFixedJW)))

# Convert SVI-JW parameters back to raw SVI parameters
vogtFixedParams <- jwToSvi(vogtFixedJW)
print(vogtFixedParams)
```

```
          vt       psit        pt        ct    varmint texp
1 0.01742625 -0.1752111 0.6997381 1.3167982 0.01162490    1
2 0.01742625 -0.1752111 0.6997381 0.3493158 0.01548182    1
            a          b       sig        rho          m
1 0.007740912 0.06924203 0.1186078 -0.3340365 0.04203375
```

Numerical optimization

- In this particular case, choosing the objective function as the sum of squared option price differences plus a large penalty for butterfly arbitrage, we arrive at the following "optimal" choices of the call wing and minimum variance parameters that still ensure no butterfly arbitrage:

$$\left(c_t, \widetilde{v}_t\right) = (0.8564763, 0.0116249)\,.$$

- Note that the optimizer has left $\widetilde{v}_t$ unchanged but has decreased the call wing.
- The resulting smiles and plots of the function g are shown in Figure 4.

In [14]:
```
%%R

# Now find values of ct and varmint that minimize distance from the original curve
strikes <- (-10:10)/10;

callVals <- function(sviparams,k){
  BSFormula(1, exp(k), 1, 0, sigma=sqrt(svi(sviparams,k)))
}

callValsVogt <- callVals(vogtParams,strikes);

obj <- function(par){
  c <- par[1]
  varmin <- par[2]
  vogtNewJW <- vogtJW
  vogtNewJW$ct <- c
  vogtNewJW$varmint <- varmin
  vogtNew <- jwToSvi(vogtNewJW)
  callValsNew <- callVals(vogtNew,strikes)
  # We need to also heavily penalize negative g
  arbVal<- sviSliceArbitrageCompute(vogtNew)$gArb

  return(sum((callValsNew-callValsVogt)^2)+100*arbVal^2)

}

obj(par=c(vogtJW$ct,vogtJW$varmint))

upperC=max(vogtJW$ct,vogtFixedJW$ct)
lowerC=min(vogtJW$ct,vogtFixedJW$ct)
upperV=max(vogtJW$varmint,vogtFixedJW$varmint)
lowerV=min(vogtJW$varmint,vogtFixedJW$varmint)


res <- optim(par=c(vogtJW$ct,vogtJW$varmint),
             fn=obj,
             method="L-BFGS-B",
             lower=c(lowerC,lowerV),
             upper=c(upperC,upperV))

vogtOptimalJW <- vogtJW
print(vogtOptimalJW$ct <- res$par[1])
#[1] 0.8564763
print(vogtOptimalJW$varmint <- res$par[2])
#[1] 0.0116249

(vogtOptimalParams <- jwToSvi(vogtOptimalJW))
#            a         b       sig       rho         m
#1 -0.03051988 0.1027168 0.4123978 0.1007176 0.2723441
```

```
[1] 0.8564762
[1] 0.0116249
            a         b       sig       rho        m
1 -0.03051987 0.1027168 0.4123978 0.1007175 0.272344
```

The Vogt smile fixed

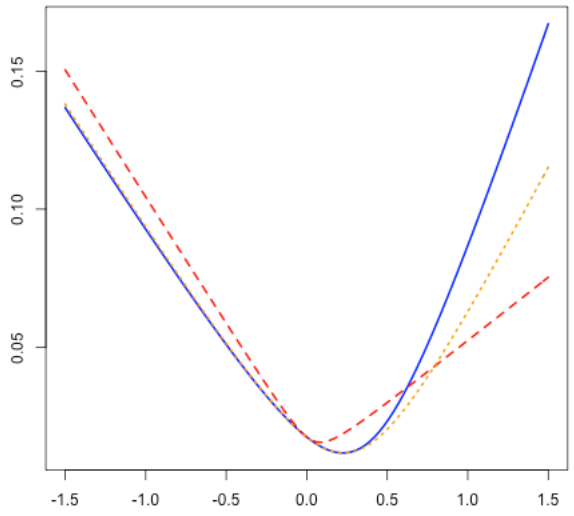
Now we have found the "optimal" choices $\left(c_t, \widetilde{v}_t\right) = (0.8564763, 0.0116249)$, let's plot the resulting smile:

In [15]:
```
%%R

# Plot Vogt smile (smileSVI)

r <- 1.5
curve(svi(vogtParams,x),from=-r,to=r,col="blue",lwd=2,
      ylab=NA,xlab=NA)
ylab=expression(w=sigma^2*T)
curve(svi(vogtFixedParams,x),from=-r,to=r,col="red",lwd=2,lty=2,add=T)
curve(svi(vogtOptimalParams,x),from=-r,to=r,col="orange",lwd=2,lty=3,add=T)

# Plot graphs of g (ggraphSVI)
curve(g(vogtParams,x),from=-r,to=r,col="blue",n=500,lwd=2,ylab=NA,xlab=NA)
lines(c(-r,r),c(0,0),lty=2);
curve(g(vogtFixedParams,x),from=-r,to=r,col="red",n=500,add=T,lwd=2,lty=2)
curve(g(vogtOptimalParams,x),from=-r,to=r,col="orange",n=500,add=T,lwd=2,lty=3)
```



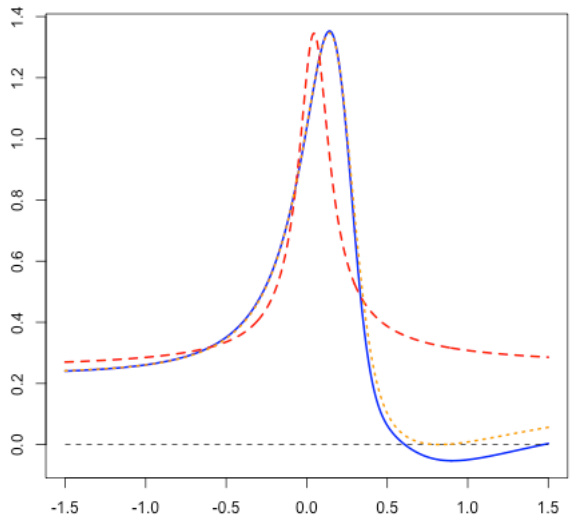


Figure 4: Plots of the total variance smile and the function g. The graphs corresponding to the original Axel Vogt parameters is solid, to the guaranteed butterfly-arbitrage-fr parameters dashed, and to the “optimal” choice of parameters dotted.

Why extra flexibility may not help

- The additional flexibility potentially afforded to us through the parameter α_t of Theorem 4.3 sadly does not help us with the Vogt smile.
- For α_t to help, we must have $\alpha_t > 0$; it is straightforward to verify that this translates to the condition $v_t\,(1-\rho^2) < \tilde{v}_t$ which is violated in the Vogt case.

In [16]:
```
%%R
print(c(vogtJW$vt*(1-vogtParams$rho^2), vogtJW$varmint))
```

```
[1] 0.01579453 0.01162490
```

Generating implied vols

- We source raw index option price data from OptionMetrics In this case, the data file is spxOptionMetrics.rData.
- We use the function $generateOptionMetricsIvols$ from optionMetricsToIVols.R to generate $spxIvols050915$.
- We use the function $plotIvols$ from plotIvols.R to plot the results.

```
spxIvols050915  <- generateOptionMetricsIvols(spxData050915)
plotIvols(spxIvols050915)
```

Raw data

Raw OptionMetrics price data looks like this:

In [17]:
```
%%R

load("spxOptionMetrics.rData")

head(spxData050915)
```

```
        secid     date symbol symbol_flag   exdate last_date cp_flag
545518 108105 20050915 SPB.FT           0 20060617  20050908       C
545519 108105 20050915 SPB.LT           0 20051217  20050809       C
545520 108105 20050915 SPB.RT           0 20060617  20050830       P
545521 108105 20050915 SPB.XT           0 20051217  20050902       P
545522 108105 20050915 SPL.FA           0 20070616        NA       C
545523 108105 20050915 SPL.FJ           0 20070616        NA       C
       strike_price best_bid best_offer volume open_interest optionid cfadj
545518      1600000      0.0        0.5      0            63 23785632     1
545519      1600000      0.0        0.5      0           170 23956762     1
545520      1600000    340.9      342.9      0           683 23785633     1
545521      1600000    361.5      363.5      0           705 23956761     1
545522       700000    538.3      541.3      0             0 26643070     1
545523       850000    406.1      409.1      0             0 26643074     1
       ss_flag root suffix ticker index_flag             issuer div_convention
545518       0  SPB     FT    SPX          1 CBOE S&P 500 INDEX               I
545519       0  SPB     LT    SPX          1 CBOE S&P 500 INDEX               I
545520       0  SPB     RT    SPX          1 CBOE S&P 500 INDEX               I
545521       0  SPB     XT    SPX          1 CBOE S&P 500 INDEX               I
545522       0  SPL     FA    SPX          1 CBOE S&P 500 INDEX               I
545523       0  SPL     FJ    SPX          1 CBOE S&P 500 INDEX               I
       exercise_style am_set_flag
545518              E           1
545519              E           1
545520              E           1
545521              E           1
545522              E           1
545523              E           1
```

Note that the strike prices are oiut by a factor of 1,000! So let's fix them:

In [18]:
```
%%R
spxData050915$strike_price <- spxData050915$strike_price/1000
```

Implied volatility computation for index options

- We compute all implied volatilities from option price data
  - We don't need external estimates of interest rates and dividends
- We use put-call parity to get implied forward prices and discount factors.
  - Find the unique forward price and discount factor that minimize implied forward pricing errors.
- In this way, we can avoid errors due to non-synchronous parameter estimates and typically generate very smooth implied volatility curves.

Detail of forward and PV estimation

We choose the forward and the PV factor for each expiration so as to minimize the squared error in the put-call parity relationship:

```
        pvGuess <- 1;
        fGuess <- mean(imid+strikes);
        nearTheMoneyStrikes <- strikes[order(abs(imid))][1:6];

        include <- (strikes%in%nearTheMoneyStrikes);
        obj <- function(params){
          f <- params[1]; pv <- params[2];
          ifit <- pv*(f-strikes); # This is a vector length nk
          errmid <- (ifit-imid)*include;
          return(sum(errmid^2));
        }
        fit <- optim(c(fGuess,pvGuess),obj,method="L-BFGS-B",
                                           lower=c(min(strikes),0.5),upper=c(max(strikes),2));
        ffit <- fit$par[1];
    pvfit <- fit$par[2];
```

Implied volatility output

The resulting implied volatility output looks like this:

In [19]:
```
%%R
spxIvols050915  <- generateOptionMetricsIvols(spxData050915)
spxIvols050915[30:50,]
```

```
     Expiry        Texp Strike        Bid        Ask      Fwd     CallMid
30 20050917 0.005475702   1150         NA 0.37535440 1227.798          NA
31 20050917 0.005475702   1155         NA 0.32547806 1227.798          NA
32 20050917 0.005475702   1160 0.30490494 0.37678130 1227.798 67.95133827
33 20050917 0.005475702   1165         NA 0.32676025 1227.798          NA
34 20050917 0.005475702   1170 0.26360852 0.34657446 1227.798 58.00243316
35 20050917 0.005475702   1175 0.24285638 0.26499148 1227.798 52.87469594
36 20050917 0.005475702   1180 0.24263809 0.28715960 1227.798 48.00243316
37 20050917 0.005475702   1185         NA 0.23352259 1227.798          NA
38 20050917 0.005475702   1190 0.19745139 0.20977679 1227.798 37.92579083
39 20050917 0.005475702   1195 0.15864560 0.19484019 1227.798 32.92579083
40 20050917 0.005475702   1200 0.16142793 0.17666020 1227.798 28.00243316
41 20050917 0.005475702   1205 0.15018761 0.15579053 1227.798 23.07907550
42 20050917 0.005475702   1210 0.12791572 0.13656493 1227.798 18.15571784
43 20050917 0.005475702   1215 0.09898340 0.10967283 1227.798 13.18126528
44 20050917 0.005475702   1220 0.08676468 0.09128081 1227.798  8.51338207
45 20050917 0.005475702   1225 0.06220211 0.08685587 1227.798  4.33090031
46 20050917 0.005475702   1230 0.06835615 0.08318531 1227.798  1.78832116
47 20050917 0.005475702   1235 0.06703419 0.07488402 1227.798  0.43430657
48 20050917 0.005475702   1240 0.08106528 0.09051137 1227.798  0.20437956
49 20050917 0.005475702   1245 0.10671125 0.15155820 1227.798  0.43430657
50 20050917 0.005475702   1250 0.11062406 0.12265544 1227.798  0.07664234
```

Implied volatility smiles: September 15, 2005

Finally plot the smiles:

In [20]:
```
%%R
res05 <- plotIvols(spxIvols050915)
```

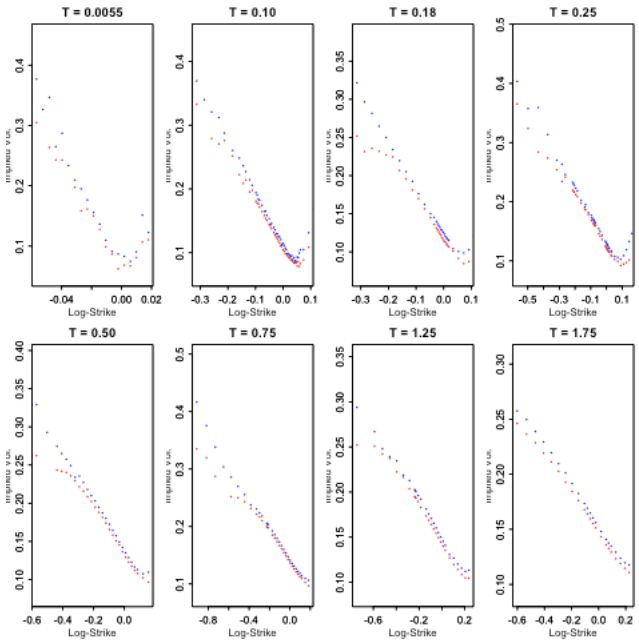


Figure 5: Here are the SPX implied volatility smiles corresponding to the output

Single-stock options

- That's fine for index options which are European-style. What about single-stock options which are American-style?
- We can't use the same put-call parity trick we used before to impute forwards and PV factors.
- In principle, we have to take the whole volatility surface into account when pricing American options.
- Ignoring this however, given some American option model that takes discrete dividends as an input, we can impute interest rates and borrow costs for single-stocks.

SVI parameters used in *The Volatility Surface*

Here are the SVI parameters corresponding to the SVI fit shown in Figures 3.2 and 3.3 of *The Volatility Surface*:

In [21]:
```
%%R

texp <- sort(unique(spxIvols050915$Texp))

svidata <- c(
 c(-0.0001449630, 0.0092965440, 0.0196713280, -0.2941176470,-0.0054273230),
 c(-0.000832134 , 0.024439766 , 0.069869455 , -0.299975308 , 0.02648364  ),
 c(-0.0008676750, 0.0282906450, 0.0873835580, -0.2892204290, 0.0592703000),
 c(-0.0000591593, 0.0331790820, 0.0812872370, -0.3014043240, 0.0652549210),
 c(0.0011431940 , 0.0462796440, 0.1040682980, -0.3530782140, 0.0942000770),
 c(0.0022640980 , 0.0562604150, 0.1305339330, -0.4387409470, 0.1111230690),
 c(0.0040335530 , 0.0733707550, 0.1707947600, -0.4968970370, 0.1496609160),
 c(0.0034526910 , 0.0917230540, 0.2236814130, -0.4942213210, 0.1854128490));

sviMatrix <- as.data.frame(t(array(svidata,dim=c(5,8))));
colnames(sviMatrix)<-c("a","b","sig","rho","m")

# Inspect the matrix
sviMatrix
```

```
             a           b        sig        rho            m
1 -0.0001449630 0.009296544 0.01967133 -0.2941176 -0.005427323
2 -0.0008321340 0.024439766 0.06986945 -0.2999753  0.026483640
3 -0.0008676750 0.028290645 0.08738356 -0.2892204  0.059270300
4 -0.0000591593 0.033179082 0.08128724 -0.3014043  0.065254921
5  0.0011431940 0.046279644 0.10406830 -0.3530782  0.094200077
6  0.0022640980 0.056260415 0.13053393 -0.4387409  0.111123069
7  0.0040335530 0.073370755 0.17079476 -0.4968970  0.149660916
8  0.0034526910 0.091723054 0.22368141 -0.4942213  0.185412849
```

Equivalent SVI-JW parameters.

We note that for longer expirations, JW parameters are almost independent of expiration:

In [22]:
```
%%R
sviToJw(sviMatrix,texp)
```

```
           vt        psit        pt        ct     varmint        texp
1 0.005461444 -0.02393127 2.1999949 1.1999972 0.005446544 0.005475702
2 0.011729194 -0.23199502 0.9217058 0.4963300 0.007865899 0.101300479
3 0.014634977 -0.23575331 0.7146813 0.3940218 0.008422155 0.177960301
4 0.015913797 -0.24169930 0.6783336 0.3641304 0.009867464 0.254620123
5 0.018220317 -0.24736379 0.6536137 0.3125000 0.011214052 0.503764545
6 0.019460020 -0.25260651 0.6687175 0.2608696 0.011772234 0.752908966
7 0.020901180 -0.26222826 0.6791513 0.2282609 0.011915278 1.251197810
8 0.022010229 -0.26465446 0.6984348 0.2364130 0.012168504 1.749486653
```

SVI fit example: September 15, 2005

Now check the quality of the SVI fit:

In [23]:
```
%%R
res05 <- plotIvols(spxIvols050915,sviMatrix);
```

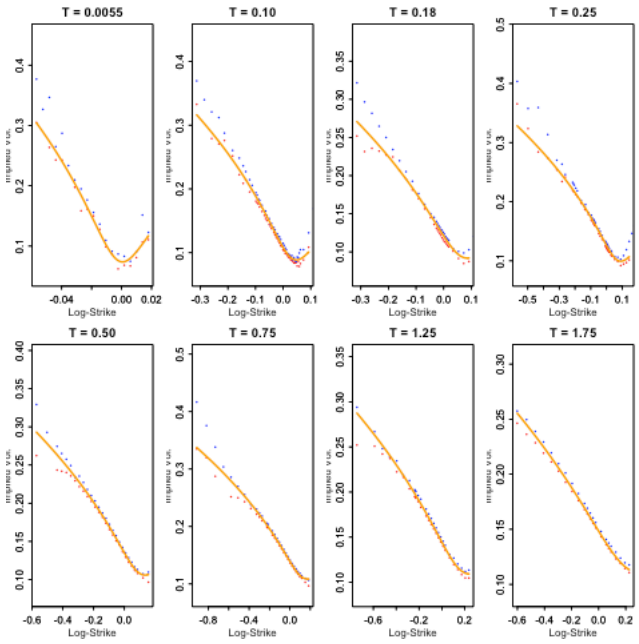


Figure 6: This fit is the one shown in Figure 3.3 of *The Volatility Surface*

Comments on the SVI fit

- It's hard to deny that this SVI fit is strikingly good.
  - At least in this particular case, the SVI parameterization gives an arbitrage-free surface with very good fits to the data.
- We would like to achieve a similar fit quality using only simple code written in R.

SPX total variance plot as of 15-Sep-2005

In [24]:
```
%%R
plotSVI(sviMatrix)
```

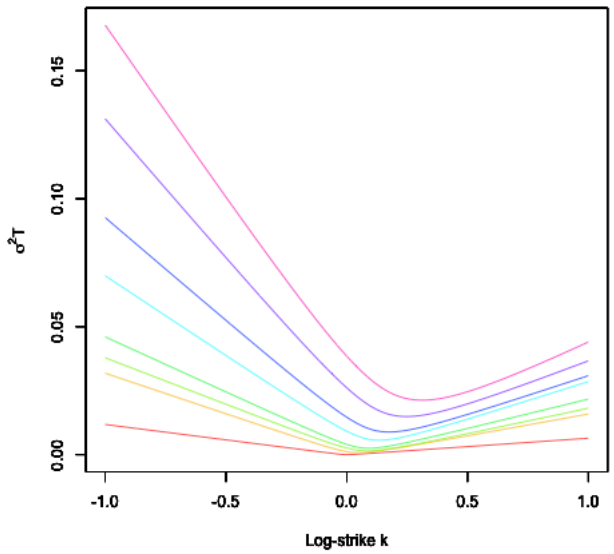


Figure 7: SVI fits to the SPX implied volatility surface as of the close on September 15, 2005, the day before triple witching.

SPX total variance plot as of 15-Sep-2005: zoomed version

In [25]:
```
%%R

plotSVI(sviMatrix,xrange=c(-.25,.25),yrange=c(0,0.01))

# Add a circle around the area of interest
y19 <- svi(sviMatrix[2,],.19)
points(x=.18,y=y19,pch=1,cex=10,pty=2,col="dark green")
```

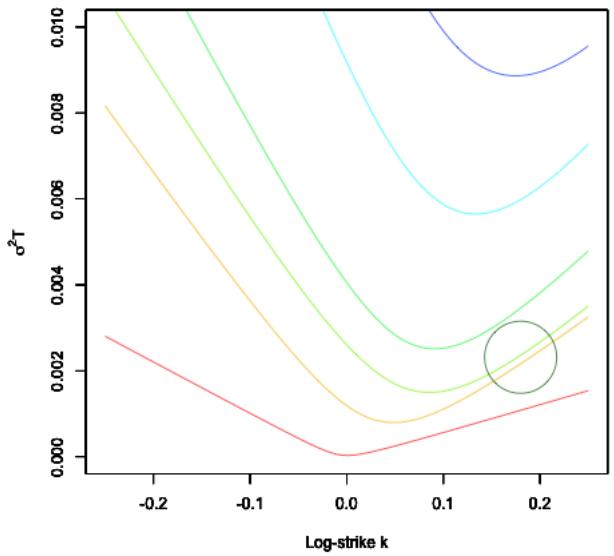


Figure 8: A zoomed view of the same plot showing lines getting very close to each other: small local variances!

The total variance plot: observations

- Note that none of the lines cross: w is a non-decreasing function of t for every log-strike k
- No calendar spread arbitrage is equivalent to no lines crossing.
  - Negative calendar spreads (lines crossing) are equivalent to negative local variances
  - Recall that (equation (1.6) on page 11 of [The Volatility Surface][1]):

$$v_{loc}(k,T) = \frac{\partial_T C}{\frac{1}{2} K^2 \partial_{K,K} C}$$

  - Lines moving together mean low local variance
  - Lines moving apart mean high local variance

Stineman interpolation

- Monotonic spline interpolation is essential. Stineman spline is a particularly attractive choice.
- Here's the term structure of total variance at constant $k = 0.19$ (red is cubic spline, blue is Stineman spline):

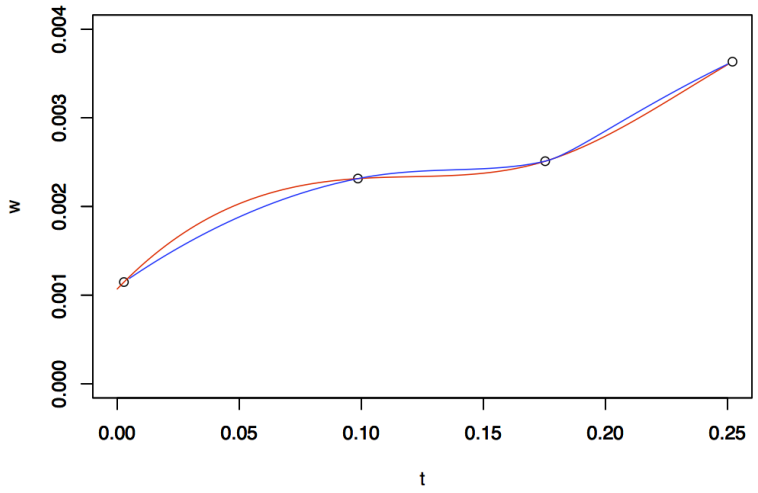


Figure 9: We see that the cubic spline generates local variances that are too small – they can even go negative.

Fitting SVI slice-by-slice

Fitting slice-by-slice is not a good idea. Let's see why:

In [26]:
```
%%R

spxOptData <- spxIvols050915

# Fit each slice individually
fit0 <- sviFit(spxOptData)
plotSVI(fit0,yrange=c(0,.1))
```

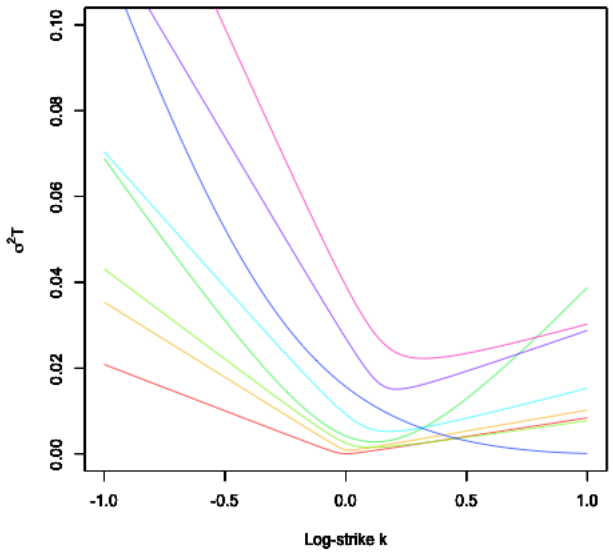


Figure 10: If we fit the slices independently, we see arbitrage everywhere!

But fits to each slice of course look good...

In [27]:
```
%%R
plotIvols(spxOptData,fit0)
```

```
$expiries
[1] 0.005475702 0.101300479 0.177960301 0.254620123 0.503764545 0.752908966
[7] 1.251197810 1.749486653

$atmVol
[1] 0.07533174 0.10928402 0.11988351 0.12567828 0.13480585 0.13956347 0.14513026
[8] 0.14926899

$atmSkew
[1] -0.7987708 -0.7411599 -0.5534874 -0.4727589 -0.3427874 -0.3040462 -0.2436678
[8] -0.2052787
```

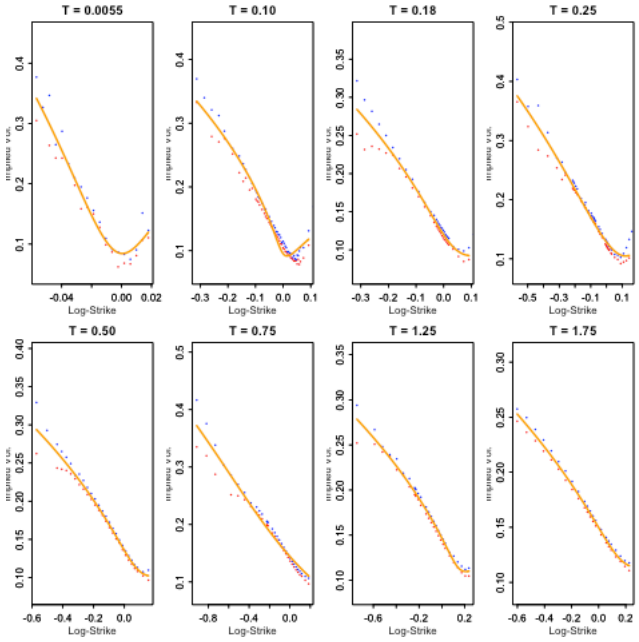


Figure 11: ...but fits to individual slices are good

Fitting with no calendar spread arbitrage

- To achieve a fit with no calendar spread arbitrage:
  - We need to find explicit constraints on the parameters.
  - We need a good initial guess.

The SVI square root parameterization

Consider the following special case of SSVI:

SVI square-root parameterization

$$w(k,\theta_t) = \frac{\theta_t}{2}\left\{1 + \rho\varphi(\theta_t)k + \sqrt{(\varphi(\theta_t)k+\rho)^2 + (1-\rho^2)}\right\}$$

with

$$\varphi(\theta) = \frac{\eta}{\sqrt{\theta}}.$$

- The SVI square-root parameterization is free of static arbitrage for up to some (very long in practice) expiration.

Square-root fit example: September 15, 2005

In [28]:
```
%%R

texp <- unique(spxOptData$Texp)
fitSqrt <- sviSqrtFit(spxOptData)
plotIvols(spxOptData,fitSqrt)
```

```
$expiries
[1] 0.005475702 0.101300479 0.177960301 0.254620123 0.503764545 0.752908966
[7] 1.251197810 1.749486653

$atmVol
[1] 0.07533174 0.10928402 0.11988351 0.12567828 0.13480585 0.13956347 0.14513026
[8] 0.14926899

$atmSkew
[1] -0.7987708 -0.7411599 -0.5534874 -0.4727589 -0.3427874 -0.3040462 -0.2436678
[8] -0.2052787
```

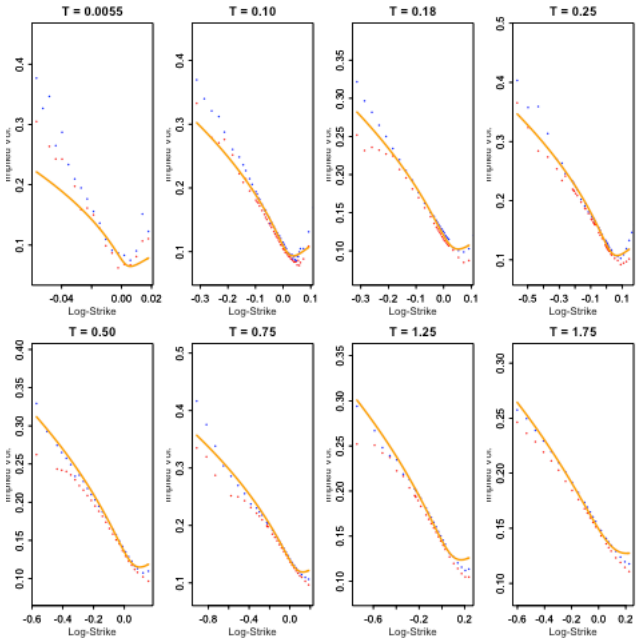


Figure 12: SVI square-root fits surprisingly well! ... except for the front month.

And here's the total variance plot:

In [29]:
```
%%R
plotSVI(fitSqrt)
```

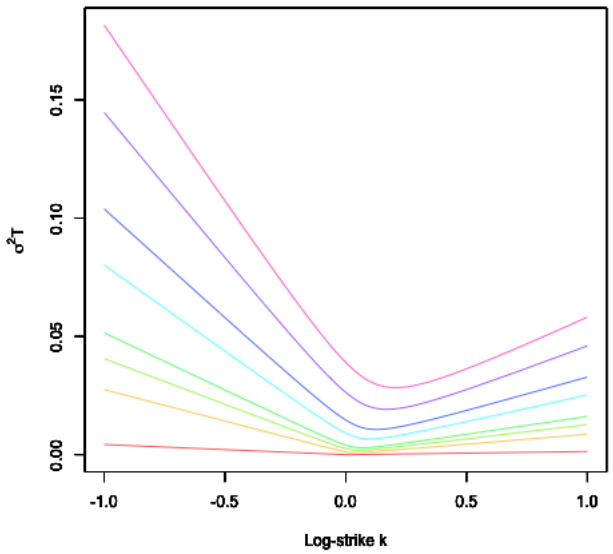


Figure 13: ... and there is no calendar spread arbitrage

Quote-driven vs order-driven markets

- In quote-driven markets, we have tight bid-offer prices for all listed options. Our fit must necessarily go through all of the bids and offers.
- In order-driven markets, it's not so clear what the current market price of an option is:
  - We may have a bid with no ask
  - Or an ask with no bid.
  - Or neither a bid nor an ask.
  - Or the bid-ask spread may just be very wide.
- In order-driven options markets therefore, the square-root fit may be more than adequate – even preferable because it has so few parameters and is therefore very robust.
- In quote-driven markets, we need to do better than the square-root fit.

Quantifying lines crossing

- Consider two SVI slices with parameters χ_1 and χ_2 where $t_2 > t_1$.
- We first compute the points k_i $(i = 1, \dots, n)$ with $n \leq 4$ at which the slices cross, sorting them in increasing order. If $n > 0$, we define the points $\widetilde{k}_i$ as
- For each of the $n + 1$ points $\widetilde{k}_i$, we compute the amounts c_i by which the slices cross:

$$c_i = \max\left[0, w(\widetilde{k}_i, \chi_1) - w(\widetilde{k}_i, \chi_2)\right].$$

Crossedness

Definition 4.4

The *crossedness* of two SVI slices is defined as the maximum of the c_i $(i = 1, , n)$. If $n = 0$, the crossedness is null.

Computing crossedness

To see if lines cross and to compute crossedness, call ComputeSviRoots from sviRoots.R. For example if we have:

In [30]:
```
%%R
# Make the lines cross
svi2 <- sviMatrix[1:2,]
svi2$a[2] <- svi2$a[2]-0.0005
plotSVI(svi2)
```

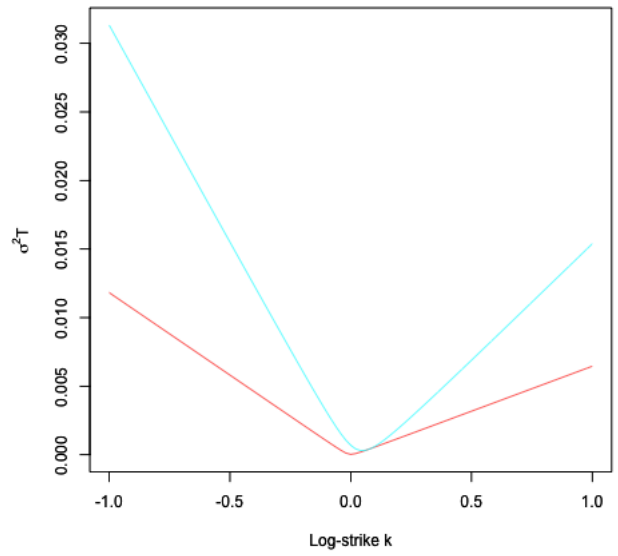


Figure 14: We lower slice 2 in order to introduce calendar spread arbitrage (crossed lines).

In [31]:
```
%%R
# Zoom in to the problem area
plotSVI(svi2,xrange=c(0,0.1),yrange=c(0,0.001))
```

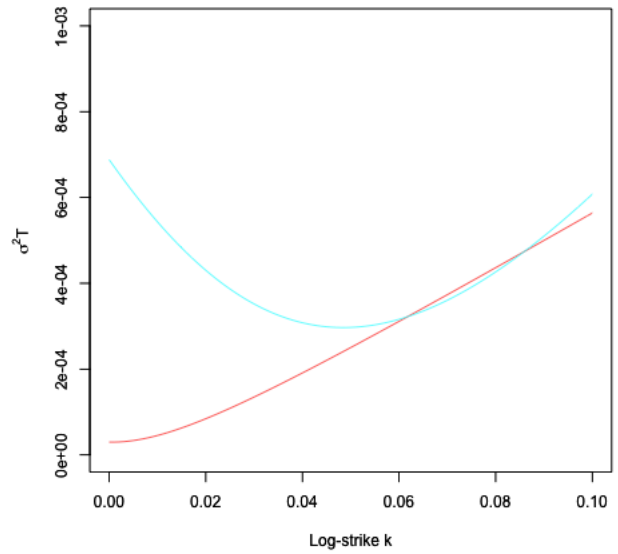


Figure 15: Zoomed version of Figure 14.

Calling sviRoots() on these two slices gives:

In [32]:
```
%%R
sviRoots(svi2)
```

```
$roots
[1] 0.06175203 0.08590506

$crossedness
[1] 1.402099e-05
```

QR fit

Finally, using *fitSqrt* as an initial guess, change SVI parameters slice-by slice, working backwards, so as to minimize the sum of squared distances between the fitted prices and the n option prices with a big penalty for crossing either the previous slice or the next slice (as quantified by the crossedness).

In [33]:
```
%%R

fitQR <- sviFitQR(spxOptData,fitSqrt)

res <- plotIvols(spxOptData,fitQR)
```

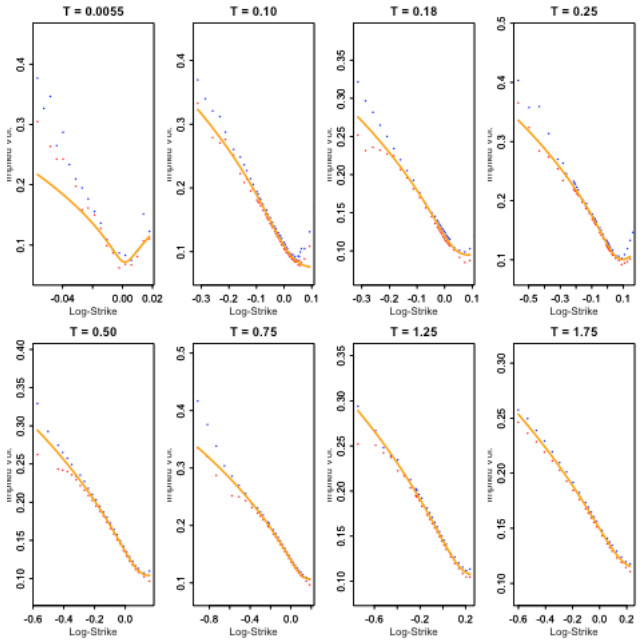


Figure 16: fitQR slice-by-slice fit quality

Let's check one of the slices:

In [34]:
```
%%R
res <- plotIvols(spxOptData,fitQR,slices=4)
```

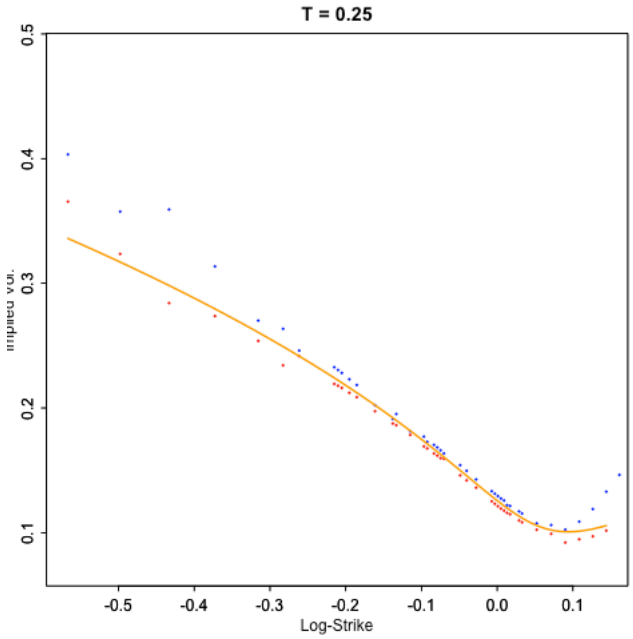


Figure 17: SPX Dec-2011 option quotes as of 3pm on 15-Sep-2011. Red triangles are bid implied volatilities; blue triangles are offered implied volatilities; the orange solid line is the Q SVI fit (*fitQR*).

And now for the total variance plot:

In [35]:
```
%%R
plotSVI(fitQR)
```

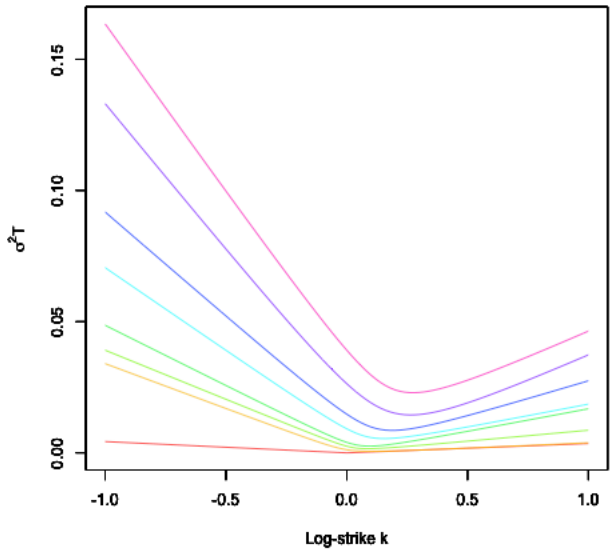


Figure 18: *fitQR* total variance plot: No calendar spread arbitrage!

However, it does look as if there are some very small local variances...

In [36]:
```
%%R
plotSVI(fitQR,yrange=c(0,0.05))
```

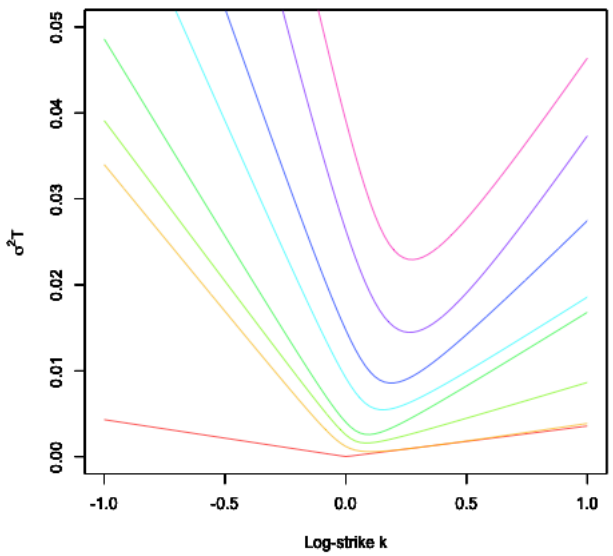


Figure 19: Closeup of the *fitQR* total variance plot: No calendar spread arbitrage, but very close...

Now we plot the empirical term structure of ATM volatility skew $\partial_k \sigma_{\mathrm{BS}}(k,T)|_{k=0}$

In [37]:
```
%%R
plot(texp,-res05$atmSkew,col="red",pch=20)
```

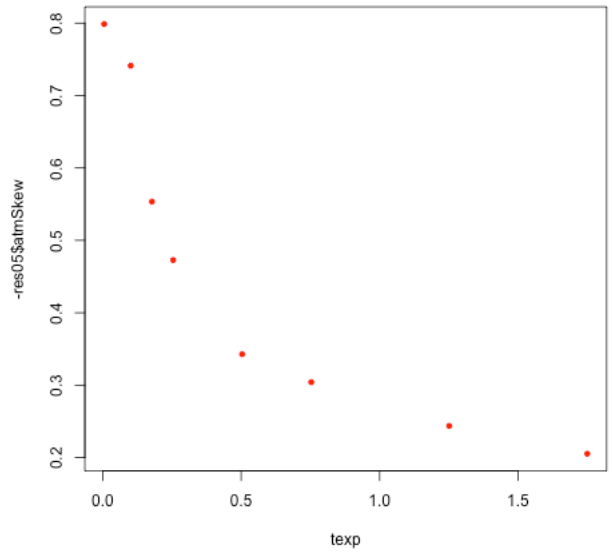


Figure 20: Empirical term structure of ATM volatility skew $\partial_k \sigma_{\mathrm{BS}}(k,T)|_{k=0}$

Now we check the various fitted surfaces for calendar spread arbitrage:

In [38]:
```
%%R

print(sviCalendarArbitrageCheck(sviMatrix)) # The pre-cooked fit
print(sviCalendarArbitrageCheck(fit0)) # The fit slice-by-slice
print(sviCalendarArbitrageCheck(fitSqrt)) # The sqrt fit
print(sviCalendarArbitrageCheck(fitQR)) # The final optimized fit
```

```
[1] 0
[1] 0.1272942
[1] 0
[1] 0
```

and then for butterfly arbitrage:

In [39]:
```
%%R

print(sviSliceArbitrageCheck(sviMatrix))
print(sviSliceArbitrageCheck(fit0))
print(sviSliceArbitrageCheck(fitSqrt))
print(sviSliceArbitrageCheck(fitQR))
```

```
numeric(0)
[1] 1
numeric(0)
numeric(0)
```

Summary of the calibration recipe

- Given mid implied volatilities $\sigma_{ij} = \sigma_{\mathrm{BS}}(k_i, t_j)$, compute mid option prices using the Black-Scholes formula.
- Fit the square-root SVI surface by minimizing sum of squared distances between the fitted prices and the mid option prices. This is now the initial guess.
- Starting with the square-root SVI initial guess, change SVI parameters slice-by slice so as to minimize the sum of squared distances between the fitted prices and the mid option prices with a big penalty for crossing either the previous slice or the next slice (as quantified by the crossedness from Definition 4.4).

Finally, let's compare the pre-cooked surface with *fitQR*:

In [40]:
```
%%R
volTVS <- function(k,t){sqrt(sviW(sviMatrix,texp,k,t)/t)}

# 3D plot of TVS volatility surface
    k <- seq(-.5,.5,0.01) # Vector of log-strikes
    t <- seq(0.04,1.74,0.02) # Vector of times
    z <- t(volTVS(k,t)); # Array of volatilities

    # Add colors
    nbcol <- 100;
    color <- rainbow(nbcol,start=.3,end=.5);
    nrz <- nrow(z)
    ncz <- ncol(z)
    # Compute the z-value at the facet centres
    zfacet <- z[-1, -1] + z[-1, -ncz] + z[-nrz, -1] + z[-nrz, -ncz]
    # Recode facet z-values into color indices
    facetcol <- cut(zfacet, nbcol);

    # Generate 3D plot
    persp(k, t, z, col=color[facetcol], phi=30, theta=30,
        r=1/sqrt(3)*20,d=5,expand=.5,ltheta=-135,lphi=20,ticktype="simple",
        shade=.8,border=NA,xlab="Log-strike k",ylab="Expiration t",zlab="Implied vol.");
```

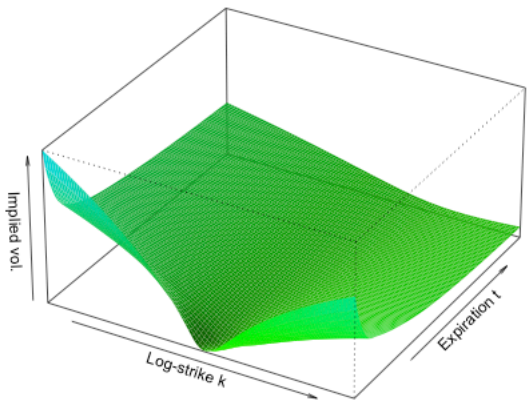


Figure 21: 3D plot of the pre-cooked SVI fit to the SPX implied volatilities as of September 15, 2005

In [41]:
```
%%R
volTVS <- function(k,t){sqrt(sviW(fitQR,texp,k,t)/t)}

# 3D plot of TVS volatility surface
    k <- seq(-.5,.5,0.01) # Vector of log-strikes
    t <- seq(0.04,1.74,0.02) # Vector of times
    z <- t(volTVS(k,t)); # Array of volatilities

    # Add colors
    nbcol <- 100;
    color <- rainbow(nbcol,start=.3,end=.5);
    nrz <- nrow(z)
    ncz <- ncol(z)
    # Compute the z-value at the facet centres
    zfacet <- z[-1, -1] + z[-1, -ncz] + z[-nrz, -1] + z[-nrz, -ncz]
    # Recode facet z-values into color indices
    facetcol <- cut(zfacet, nbcol);

    # Generate 3D plot
    persp(k, t, z, col=color[facetcol], phi=30, theta=30,
        r=1/sqrt(3)*20,d=5,expand=.5,ltheta=-135,lphi=20,ticktype="simple",
        shade=.8,border=NA,xlab="Log-strike k",ylab="Expiration t",zlab="Implied vol.");
```

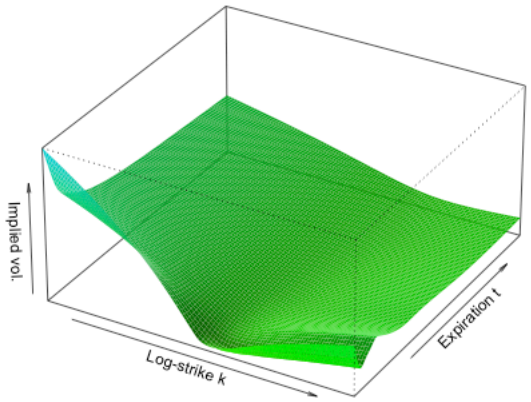


Figure 22: 3D plot of the QR fit to the SPX implied volatilities as of September 15, 2005

Note that the QR fit looks almost as good as the pre-cooked one.

Choice of objective function

- A 0.1% volatility error on an ATM option is a much bigger pricing error than on a very far out-of-the-money option.
- Practitioners take this into consideration by vega-weighting their fits.
- From the Black-Scholes formula:

$$\text{vega} = \sqrt{T}\,K\,\frac{e^{-d_2^2/2}}{\sqrt{2\,\pi}} = \sqrt{T}\,F\,\frac{e^{-d_1^2/2}}{\sqrt{2\,\pi}}$$

- Alternatively, weight by the reciprocal of the bid-ask volatility spread.

Now we repeat the whole calibration recipe for 15-Sep-2011

In [42]:
```
%%R
head(spxData110915)
```

```
         secid     date             symbol symbol_flag   exdate last_date
2559920 108105 20110915  SPX 110917C100000           1 20110917  20110815
2559921 108105 20110915 SPX 110917C1000000           1 20110917  20110915
2559922 108105 20110915 SPX 110917C1005000           1 20110917  20110829
2559923 108105 20110915 SPX 110917C1010000           1 20110917  20110829
2559924 108105 20110915 SPX 110917C1015000           1 20110917  20110912
2559925 108105 20110915 SPX 110917C1020000           1 20110917  20110909
        cp_flag strike_price best_bid best_offer volume open_interest optionid
2559920       C       100000   1106.4     1110.0      0             0 49587202
2559921       C      1000000    206.4      210.0     22          4213 49287956
2559922       C      1005000    200.2      204.1      0            11 63098880
2559923       C      1010000    195.2      199.1      0             5 63098881
2559924       C      1015000    190.2      194.1      0             5 63098882
2559925       C      1020000    186.5      190.1      0            10 54355887
        cfadj ss_flag root suffix ticker index_flag            issuer
2559920     1       0                SPX          1 CBOE S&P 500 INDEX
2559921     1       0                SPX          1 CBOE S&P 500 INDEX
2559922     1       0                SPX          1 CBOE S&P 500 INDEX
2559923     1       0                SPX          1 CBOE S&P 500 INDEX
2559924     1       0                SPX          1 CBOE S&P 500 INDEX
2559925     1       0                SPX          1 CBOE S&P 500 INDEX
        div_convention exercise_style am_set_flag
2559920              I              E           1
2559921              I              E           1
2559922              I              E           1
2559923              I              E           1
2559924              I              E           1
2559925              I              E           1
```

In [43]:
```
%%R

spxData110915$strike_price <- spxData110915$strike_price/1000
spxIvols110915 <- generateOptionMetricsIvols(spxData110915)
fitSqrt110915 <- sviSqrtFit(spxIvols110915)
fitQR110915 <- sviFitQR(spxIvols110915,fitSqrt110915)
plotIvols(spxIvols110915,fitQR110915)
```

```
$expiries
 [1] 0.005475702 0.021902806 0.041067762 0.101300479 0.177960301 0.254620123
 [7] 0.290212183 0.503764545 0.539356605 0.752908966 0.788501027 1.270362765
[13] 1.768651608 2.266940452

$atmVol
 [1] 0.1801351 0.3082525 0.2946014 0.2820179 0.2978022 0.2873252 0.2853402
 [8] 0.2792849 0.2807139 0.2758373 0.2760718 0.2684932 0.2650282 0.2637186

$atmSkew
 [1] -1.8166969 -1.3546245 -1.1209457 -0.8177976 -0.7423413 -0.5658659
 [7] -0.5458074 -0.4114060 -0.4027825 -0.3536120 -0.3710598 -0.2963508
[13] -0.2561800 -0.2448122
```

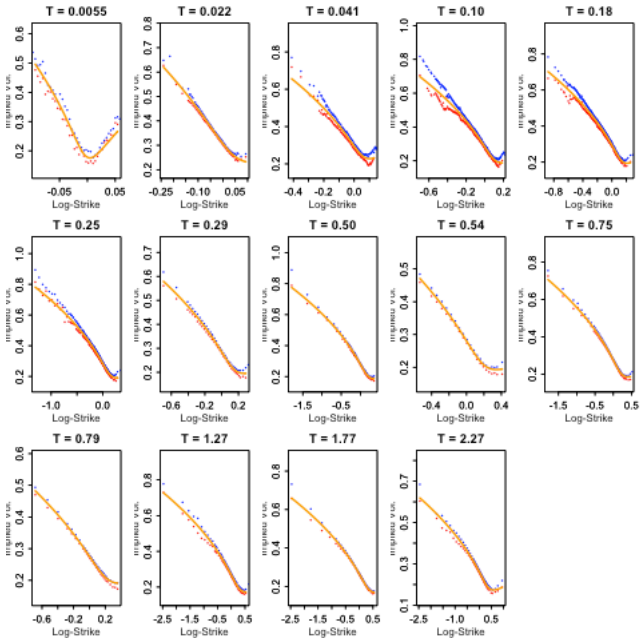


Figure 23: SPX option quotes as of 3pm on 15-Sep-2011. Red triangles are bid implied volatilities; blue triangles are offered implied volatilities; the orange solid line is the quartic ro
(QR) SVI fit.

SVI square-root calibration: December 2011 detail

In [44]:
```
%%R

foo <- plotIvols(spxIvols110915,fitQR110915,slices=7)
```

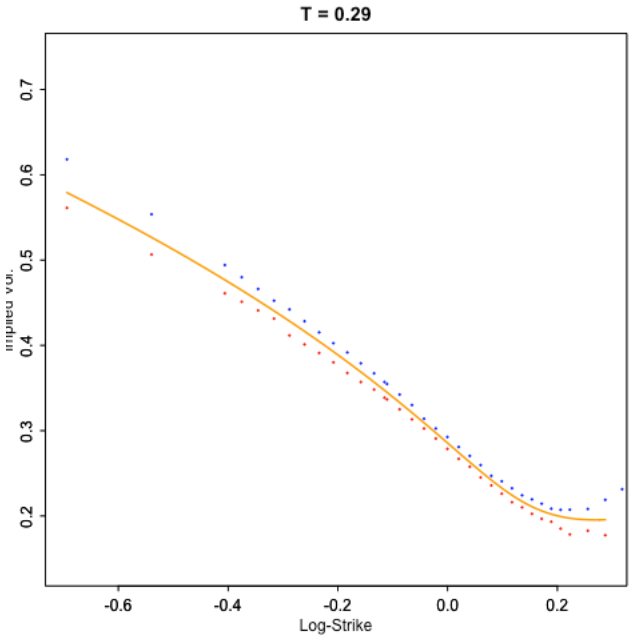


Figure 24: SPX Dec-2011 option quotes as of 3pm on 15-Sep-2011. Red triangles are bid implied volatilities; blue triangles are offered implied volatilities; the orange solid line is t quartic roots SVI fit

In [45]:
```
%%R
# Figure 20: 15-Sep-2011 SPX smile to Mar-2011
foo <- plotIvols(spxIvols110915,fitQR110915,slices=8)
```

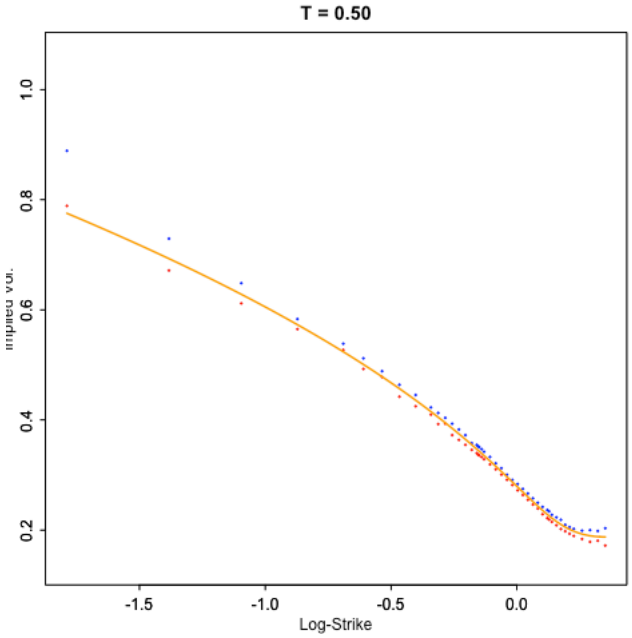


Figure 25: SPX Mar-2012 option quotes as of the close on 15-Sep-2011. Red triangles are bid implied volatilities; blue triangles are offered implied volatilities; the orange solid line is t SVI fit

Summary

- The construction we have presented shows how to fit SVI to the volatility surface whilst guaranteeing no-static-arbitrage.
- The SSVI parameterization allows us to summarize the shape of the whole volatility surface with very few parameters.
  - Invaluable in particular for analysis of volatility surface dynamics.
- The fitting code is all yours!

References

1. ^ Shalom Benaim and Peter Friz, Regular variation and smile asymptotics, *Mathematical Finance* **19**(1), 1–12, (2009).
2. ^ Gerolamo Cardano, *Ars magna or The Rules of Algebra*, Dover (1545).
3. ^ Jim Gatheral, *The Volatility Surface: A Practitioner's Guide*, John Wiley and Sons, Hoboken, NJ (2006).
4. ^ Jim Gatheral and Antoine Jacquier, Convergence of Heston to SVI, *Quantitative Finance* **11**(8) 1129–1132 (2011).
5. ^ Jim Gatheral and Antoine Jacquier, Arbitrage-free SVI volatility surfaces, *Quantitative Finance* **14**(1) 59-71 (2014).
6. ^ Roger Lee, The moment formula for implied volatility at extreme strikes, *Mathematical Finance*, **14**(3), 469–480 (2004).