

Co-addition using kernel smoothing

Martin D. Weinberg

March 20, 1996

Contents

1 Overview	1
2 Kernel smoothing theory and discussion	1
3 Application to co-addition	3
3.1 Finite pixel size	3
3.2 Procedure	3
4 Tests on Coma and M67 fields	4
A Possible choices for smoothing kernel	6
B Convolved Gaussian smoothing kernel	6
C Formula for co-added image	7

1 Overview

This is the revised version of the previous write-up. I have checked the key expressions and provided some additional discussion and examples. Figures 2 and 3 below illustrate the main point: square pixels have nothing to do with Nature and may introduce artifacts so a better result is obtained by smoothing the pixels with a two-dimensional Gaussian or tabulated PSF. The final section (§4) show some examples of optimal smoothing for Coma and M67 fields.

Key points and equations:

1. Equation (21) gives the density estimate for one frame where the weight m_j is given by equation (22) and the convolved smoothing kernel by equation (20)¹. This expression would be used for identifying features. 5

¹Tom: note the discussion of ‘dead zones’ in §B.

2. Equation (24) gives the contribution from the j^{th} frame pixel to the sub pixel with linear scale l and center x, y . This expression would be used to produce the co-added image.
3. The minimum of the MISE expression (eq. 12) would be used to find the optimal smoothing length h for the estimate in equation (21).
4. I have argued that the recovered mean ‘PSF’ already computed as part of the photometry task should be close to the ideal smoothing kernel because this ‘PSF’ represents the sky + telescope + optics convolved with the pixel profile. Without using M_1 , one can probably estimate the appropriate value of h by “fitting” equation (20) to this ‘PSF’.

2 Kernel smoothing theory and discussion

Suppose we have a one-dimensional continuous process with a well-defined but unknown distribution. We sample at discrete points and want to estimate the underlying distribution. A everyday solution is the histogram. However, we most often subjectively choose a bin size but here I want to motivate the optimal choice.

Let $f(x)$ be the underlying unknown density and $\hat{f}(x)$ be the smoothed estimate from the n events at the points x_i :

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right) \quad (1)$$

where h is the smoothing length. The smoothing kernel K in this case is the boxcar: the function which is 1 if $-1/2 < x < 1/2$ and zero otherwise. The kernel K is assumed to be unit normalized: $\int K(x)dx = 1$.

The quality of the estimate may be measured by computing the mean square error at x (MSE_x):

$$\begin{aligned} MSE_x &= E \left\{ \hat{f}(x) - f(x) \right\}^2 \\ &= \left\{ E\hat{f}(x) - f(x) \right\}^2 + \text{var} \hat{f}(x) \end{aligned} \quad (2)$$

where E denotes the expectation value after many trials. To get a global measure, we may integrate this over the entire domain to get the *mean integrated square error*:

$$\begin{aligned} MISE &= \int dx \left\{ E\hat{f}(x) - f(x) \right\}^2 + \\ &\int dx \text{var} \hat{f}(x). \end{aligned} \quad (3)$$

This tells us that the error in the estimate is comprised of two parts: 1) error due to bias in the estimate; and 2) error due to variance in the estimate. If we make h very small for fixed sample size n , the bias vanishes but the variance term increases. Similarly, if we make h big, the bias gets large while the variance term decreases. Clearly, the optimal solution is in between.

A common choice in the literature is to empirically choose the smoothing length h which minimizes the *integrated square error*:

$$ISE = \int dx \left\{ \hat{f}(x) - f(x) \right\}^2. \quad (4)$$

Sparing you the algebra (“Density Estimation” by B. W. Silverman, 1986, pg. 48), this can be expanded (with a slight approximation which is increasingly good for increasing n) to give:

$$M_1(h) = \frac{1}{n^2 h} \sum_i \sum_j K^* \left(\frac{x_i - x_j}{h} \right) + \frac{2}{nh} K(0), \quad (5)$$

where $K^*(z) = K^{[2]}(z) - 2K(z)$ and $K^{[2]}$ is the convolution of the kernel with itself. In the case of a Gaussian kernel, $K^{[2]}$ is a Gaussian with twice the variance ($\sigma^2 = 2$). We now choose the value of h which minimizes M_1 .

This is not as complicated as it sounds and is all trivially computed in 1d using Fourier transforms. All of this works for any smoothing kernel, not just the boxcar. In practice most people use Gaussian

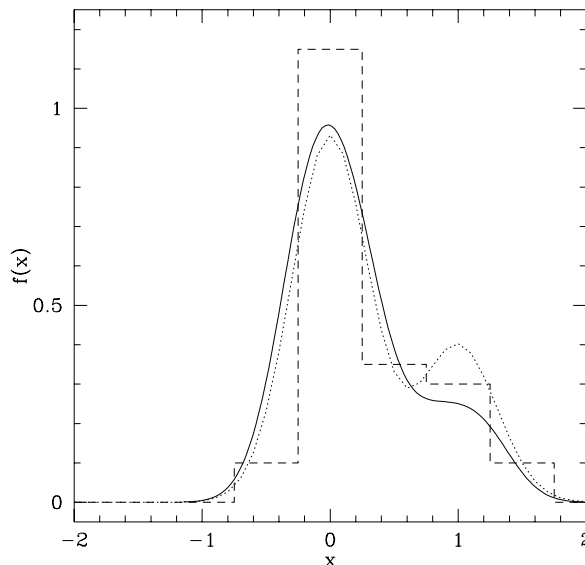


Figure 1: Density estimation using kernel smoothing using optimal h based on M_1 (solid) compared with histogram (dashed). Fifty points selected from the sum of two Gaussian distributions with means at 0 and 1 (dotted).

rather the boxcar; the Gaussian is more efficient, converging to the exact result more quickly.

I have a code using this approach called from SM which replaces histograms and works remarkably well. For example, Figure 1 shows the density estimates using this method and a histogram for 50 points randomly chosen from the sum of two Gaussians with means at 0 and 1 and $\sigma = 0.3$ with relative amplitudes in the ratio 7:3. The histogram bin size is chosen according to the standard rule: $\log_2(50)$ bins spanning the data domain. The kernel smoothing approach yields good estimate of underlying distribution, although the amplitude of the second peak is underestimated. Also note that linear interpolation is similar to connecting the histogram centers by lines, which yields a rather worse estimate than the kernel estimate.

3 Application to co-addition

3.1 Finite pixel size

One might assert that our camera pixels sample the true flux distribution and assign the photons uniformly in the pixel area (the ‘maximum entropy’ assumption). Doing this in one-dimension for simplicity, we smooth with the new kernel:

$$\bar{K}_j(x; L, h) = \frac{1}{Lh} \int_{x_j-L/2}^{x_j+L/2} dy K\left(\frac{x-y}{h}\right) \quad (6)$$

where L is the linear size of the *active* pixel, x_j is the pixel center. The pixel size and smoothing kernel are not explicit parameters. The convolved kernel \bar{K} is also unit normalized as you can see by explicitly integrating over all x . As $h \rightarrow 0$, K/h approaches a delta function which makes \bar{K} into the unit normalized boxcar function with width L . The estimate for $f(x)$ is now:

$$\hat{f}(x) = \frac{1}{n} \sum_{j=1}^{n_{pix}} m_j \bar{K}_j(x) \quad (7)$$

where n_{pix} is the number of pixels in a frame², the m_j is the flux (proportional to the number of counts) in pixel j and $n = \sum_j m_j$. The MISE measure, M_1 , is also similar:

$$M_1(h) = \frac{1}{n^2} \sum_{i=1}^{n_{pix}} \sum_{j=1}^{n_{pix}} m_i m_j \mathcal{K}^*(x_i - x_j) + \frac{2}{n} K(0). \quad (8)$$

This is derived by applying the ‘Maximum Entropy’ assumption to equation 5 and the resulting expression is related to the coadd formula given in Appendix C³. Even though the first term scales as n_{pix}^2 , in practice, one can truncate the sum for $|x_i - x_j| < Nh$ for some small integer N . Since h is probably a fraction of a pixel (true seeing half-width), Nh will probably be 1.5 to 2 pixels for a Gaussian smoothing kernel. It is possible and maybe better to use a polynomial smoothing kernel with compact support. Both

²For this one-dimensional example, our frame is only one column or row of pixels.

³Mathematically, the function $\mathcal{K}(x_i - x_j)$ is equal to the contribution of $\bar{K}_j(x)$ to the pixel centered at x_i . In two dimensions, this is related to the coadd formula given in Appendix C as follows: $\mathcal{K}(x_i - x_j, y_i - y_j) = C(x_i - x_j, y_i - y_j; L_{true}, L_{active}, h)$ where L_{true} is the distance between pixel centers and L_{active} is the size of the photon-sensitive pixel. Therefore, $\mathcal{K}^*(\cdot, \cdot) = C(\cdot, \cdot; \cdot, \cdot, \sqrt{2}h) - 2C(\cdot, \cdot; \cdot, \cdot, h)$

choices will be discussed in the Appendix with explicit expression for the Gaussian.

Finally, everything said so far carries directly to two dimensions. The two-dimensional smoothing kernel, \bar{K} is:

$$\bar{K}_j(x, y) = \frac{1}{L^2 h^2} \int_{x_j-L/2}^{x_j+L/2} \int_{y_j-L/2}^{y_j+L/2} du dv \times K\left(\frac{x-u}{h}, \frac{y-v}{h}\right) \quad (9)$$

$$(10)$$

where (x_j, y_j) is the center of the j^{th} pixel. The estimate is now:

$$\hat{f}(x, y) = \frac{1}{n} \sum_{j=1}^{n_{pix}} m_j \bar{K}_j(x, y) \quad (11)$$

with

$$M_1(h) = \frac{1}{n^2} \sum_{i=1}^{n_{pix}} \sum_{j=1}^{n_{pix}} m_i m_j \mathcal{K}^*(x_i - x_j, y_i - y_j) + \frac{2}{n} K(0, 0). \quad (12)$$

3.2 Procedure

1. Choose kernel and convolve to get \bar{K} following equation (10). The efficiency does not strongly depend on the kernel as long as it the function is reasonable (e.g. rapidly falling tails, see Silverman op. cit.). This suggests that we use a convenient kernel. For a Gaussian kernel, the second of equation (10) applies.
2. Using equation (12), estimate h by minimizing M_1 over a bunch of frames. This characterization will mostly likely only need to be performed once or infrequently at worst.
3. For each frame in the co-add, use the estimate from equation (11) to determine the contribution to the underlying flux. If the current sub-pixel is within Nh of a bad pixel on a particular frame, the contribution from the frame is ignored ($N = 3, 4$? This 3 or 4 ‘sigma’ value of N will probably correspond to between 1 and 2 pixels lengths. A compact kernel give this cut explicitly. See Appendix for more details).

- Finally, if one wants the new image (e.g. for the co-added image product, the flux estimate \hat{f} must be integrated over the new pixel locations. This expression is analytic in general if the kernel is analytic. Examples in Appendix.

This scheme should preserve flux and from my own experience with one-dimensional estimations, minimize spurious effects from bin geometry. Because $h \ll L$ (see §4), the convolution described by the kernel \bar{K} is very compact and therefore relatively cheap to compute (confirmed by Gene Kopan). In addition, M_1 provides a statistical check on the quality of the smoothing (although this would be used as a spot check once h was determined a priori). Tests so far suggest that the optimal h will be small (see next section).

4 Tests on Coma and M67 fields

Here we describe the results of evaluating M_1 from equation (12) to 2MASS protocluster data from Coma⁴ and M67⁵. Recall that M_1 is derived by evaluating the MISE for the estimate:

$$\begin{aligned} MISE &= \int dx dy \{ \hat{f} - f \}^2 \\ &= \int dx dy \{ \hat{f}^2 - 2\hat{f}f + f^2 \} \quad (13) \end{aligned}$$

where \hat{f} is the estimate and f is the true flux density. Of course, we do not know the true f , but by taking the expectation of MISE, we can evaluate the first two terms from the data and these make up M_1 . The last term is an unknown constant but it does not depend on h and is therefore irrelevant to the minimization. However, by estimating $\int f^2$ directly from the pixelated data, we can add back $\int f^2$ to M_1 and interpret the resulting expression as a MISE. The square root of the MISE tells indicates the RMS deviation of the estimate from the true value. I will plot this RMS value rather than M_1 in what follows.

Figure 2 shows a randomly selected field from a Coma (RA = 12:57:24.16, DEC = +27:54:12.5). The RMS is quite increases rapidly as $h \rightarrow 0$. This is consistent with my assertion in §1 that square pixels

⁴Test frames supplied by Tom Chester

⁵Test frames supplied by Mike Skrutskie

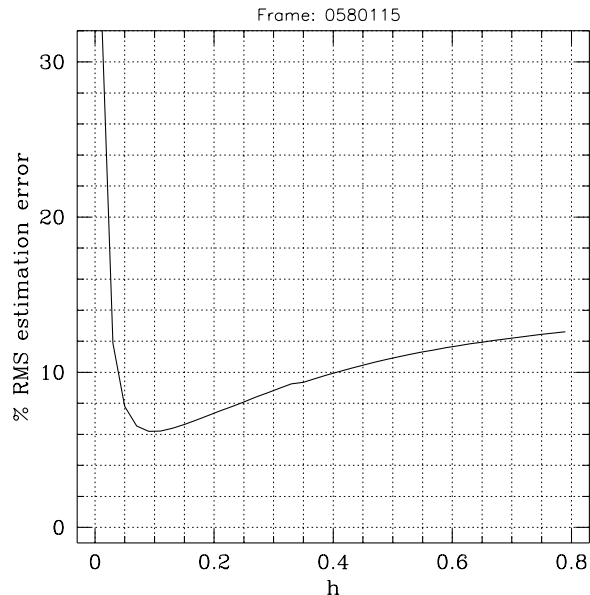


Figure 2: Run of RMS estimation error (derived from M_1) with h for Coma field.

have nothing to do with Nature. Surprisingly, the RMS drops quickly with increasing h and has a minimum at $h \approx 0.1$ and grows slowly for larger h . The ‘glitch’ at $h = 0.33$ is an artifact from my truncating the convolution at $3h$ pixels. So beyond $h = 1/3$, the convolution includes a larger domain and is a bit more accurate for $h > \sim 3.3$. Sorry for being skimpy.

Figure 3 shows this same field with others separated by 10 frames on the same scan. Each has roughly the same shape. A few of these have even smaller minima and steeper profiles. In each of these cases, there is a very bright source in the frame.

This motivated checking M_1 on point sources dominated frames e.g. M67 (Fig. 4). Note the same steep profile seen in the Coma frames with bright sources. The overall magnitude of the RMS is smaller indicating that the smoothing algorithm gives a better overall estimate for points sources. This is nicely illustrated by comparing a single raw M67 frame to its kernel smoothed counterpart with $h = 0.1$ (Figures 5 and 6).

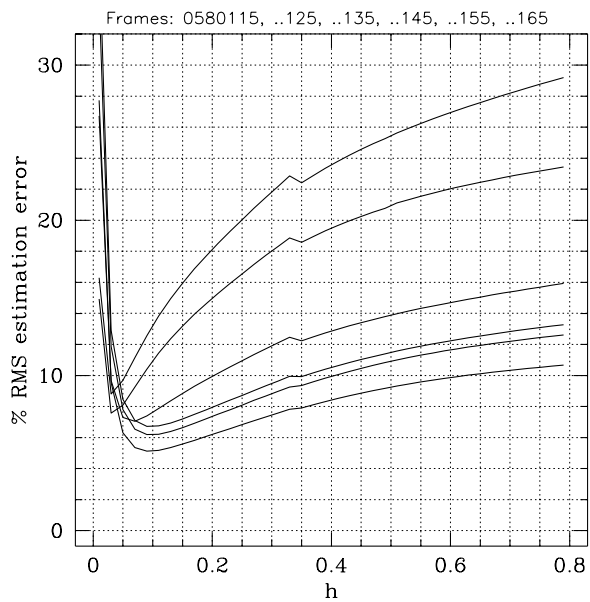


Figure 3: RMS estimation error for a number of randomly selected Coma fields.

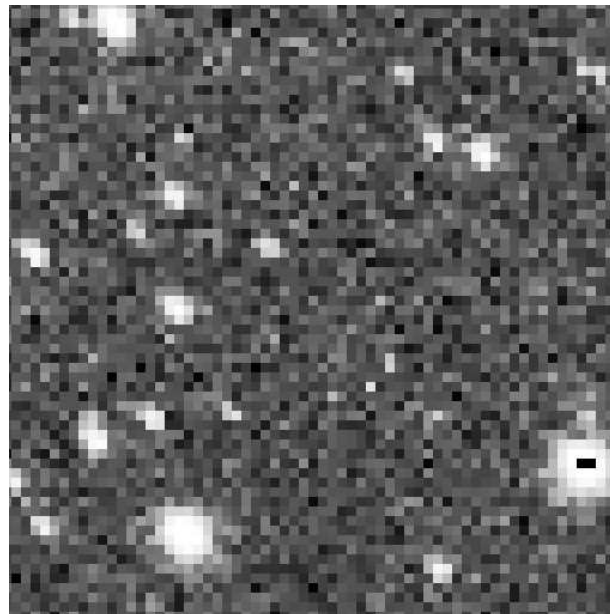


Figure 5: Fragment of the raw M67 frame used in the PR viewgraph.

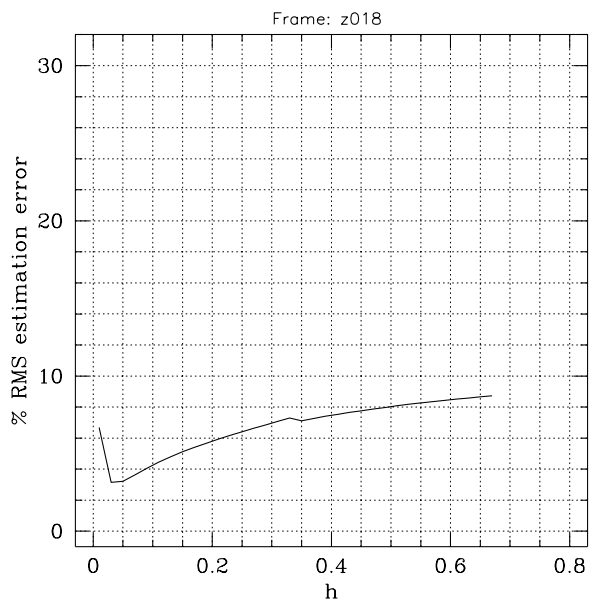


Figure 4: RMS estimation error for an M67 frame

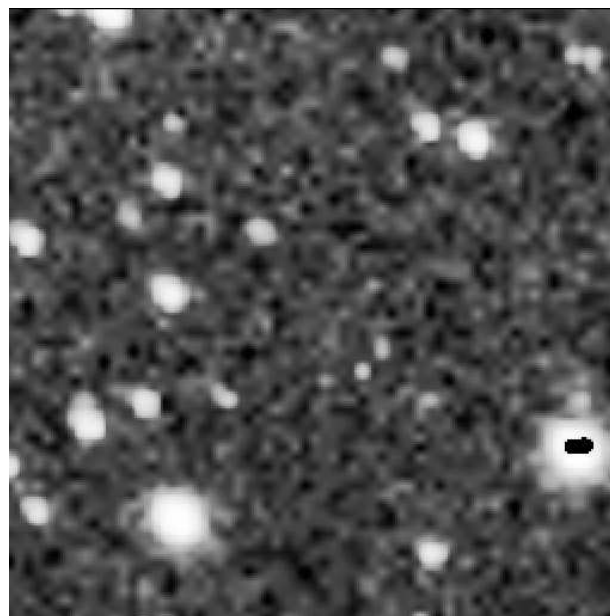


Figure 6: As in Fig. 5 but kernel smoothed with $h = 0.1$. Note that the optical double in the upper right is resolved with a single frame, optimally smoothed.

A Possible choices for smoothing kernel

Some standard choices in one-dimension for K are the Gaussian,

$$K_g(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \quad (14)$$

the simple polynomial due to Epanechnikov,

$$K_e(x) = \begin{cases} \frac{3}{4}(1-t^2) & \text{for } |t| < 1, \\ 0 & \text{otherwise,} \end{cases} \quad (15)$$

and the so-called ‘biweight’ function,

$$K_e(x) = \begin{cases} \frac{15}{16}(1-t^2)^2 & \text{for } |t| < 1, \\ 0 & \text{otherwise.} \end{cases} \quad (16)$$

Formally, the Epanechnikov is the most efficient among these.

In d dimensions these become

$$K_g(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}} e^{-\mathbf{x}^T \mathbf{x}/2}, \quad (17)$$

the simple polynomial due to Epanechnikov,

$$K_e(\mathbf{x}) = \begin{cases} \frac{d+2}{2c_d}(1-\mathbf{x}^T \mathbf{x}) & \text{for } \mathbf{x}^T \mathbf{x} < 1, \\ 0 & \text{otherwise,} \end{cases} \quad (18)$$

where c_d is the volume of the d -dimensional sphere ($c_1 = 2, c_2 = \pi$), and the so-called ‘biweight’ function for $d = 2$,

$$K_e(\mathbf{x}) = \begin{cases} \frac{3}{\pi}(1-\mathbf{x}^T \mathbf{x})^2 & \text{for } \mathbf{x}^T \mathbf{x} < 1, \\ 0 & \text{otherwise.} \end{cases} \quad (19)$$

In the discussion below, I will use equation (17) because it is easier to write down closed form expressions. Expressions using equation (18) are closed form but has multiply-branched condition statements due to the discontinuity at its edge.

B Convolved Gaussian smoothing kernel

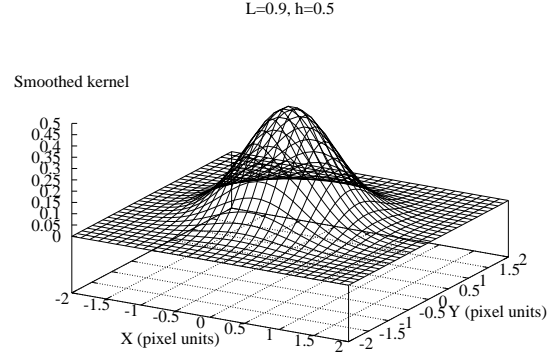


Figure 7: Gaussian kernel with $h = 0.5$ convolved over pixel with active linear size $L = 0.8$. The wire-frame diagram shows the contribution of the flux in pixel centered about the origin at arbitrary x, y point.

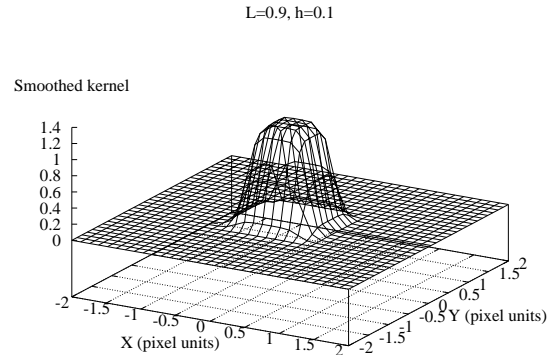


Figure 8: Same as Fig. 7 but with $h = 0.1$. As $h \rightarrow 0$, the smoothing kernel would be a box reflecting the input pixel response.

The two-dimensional smoothing kernel follows directly by substituting equation (17) in equation(10):

$$\begin{aligned} \bar{K}_j(x, y; L, h) = & \frac{1}{4L^2} \left[\operatorname{erf} \left(\frac{x - x_j + L/2}{\sqrt{2}h} \right) - \right. \\ & \left. \operatorname{erf} \left(\frac{x - x_j - L/2}{\sqrt{2}h} \right) \right] \times \\ & \left[\operatorname{erf} \left(\frac{y - y_j + L/2}{\sqrt{2}h} \right) - \right. \\ & \left. \operatorname{erf} \left(\frac{y - y_j - L/2}{\sqrt{2}h} \right) \right] \end{aligned} \quad (20)$$

where the density estimate is

$$\hat{f}(x, y) = \frac{1}{n} \sum_{j=1}^{n_{pix}} m_j \bar{K}_j(x, y) \quad (21)$$

with

$$n = \sum_j m_j. \quad (22)$$

Noting that

$$\int dx \operatorname{erf}(x/a) = \frac{a}{\sqrt{\pi}} e^{-x^2/a^2} + x \operatorname{erf}(x/a) \quad (23)$$

one can easily verify, as a check, that this expression is unit normalized. Just as in the one-dimensional case, it is easy to see that $\bar{K}_j(x, y; L, h)$ looks like a two-dimensional boxcar (i.e. the pixel itself) as $h \rightarrow 0$. Note that L is the size of the active pixel and x_j, y_j is the pixel center. Because of the dead ‘gutters’ of width δ , the distance between centers, $|x_j - x_{j-1}|$, is $L + \delta$.

Figures 7 and 8 give two examples of these convolved kernels using equation (20).

C Formula for co-added image

Given a pixel of side length l and center at x, y , the contribution from the frame pixel j follows by integrating equation (20) using equation (23):

$$\begin{aligned} C(x, y; l, L, h) = & \frac{1}{4L^2} \left\{ g_{h,l}(x + L/2 - x_j) - \right. \\ & \left. g_{h,l}(x - L/2 - x_j) \right\} \times \\ & \left\{ g_{h,l}(y + L/2 - y_j) - \right. \\ & \left. g_{h,l}(y - L/2 - y_j) \right\} \end{aligned} \quad (24)$$

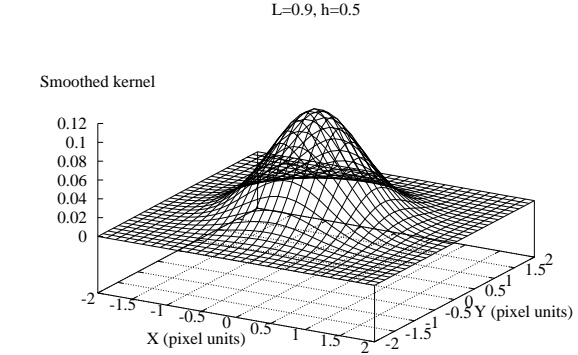


Figure 9: Smoothing kernel with $h = 0.5$ convolved over pixel with $L = 0.8$ and integrated over a sub pixel with linear size $l = 0.5$. The axes show the sub pixel relative the center of the frame pixel.

where

$$g_{h,l}(x) = z_h(x + l/2) - z_h(x - l/2), \quad (25)$$

and

$$z_h(x) = \sqrt{\frac{2}{\pi}} h e^{-x^2/2h^2} + x \operatorname{erf}(x/\sqrt{2}h). \quad (26)$$

Figures 9 and 10 give two examples of the contributions from the density estimation of one frame pixel to sub-sampled pixels (using eq. 24).

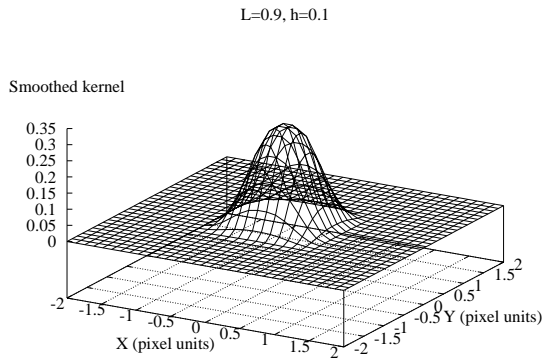


Figure 10: Same as Fig. 9 but with $h = 0.1$.

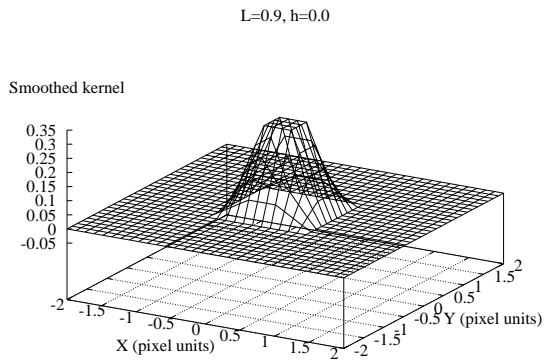


Figure 11: Same as Fig. 9 but with $h = 0$. The contribution is one quarter of the total as long as the sub pixel is inside of the frame pixel. As the sub pixel moves outside the frame pixel, the contribution decreases linearly to zero.