Background

- Mobile/embedded devices are everywhere affecting
  - Telecommunications
  - Consumer electronics
  - Household appliances
- Embedded image analysis algorithms include
  - Contrast enhancement
  - Mobile panorama
  - Face recognition
  - Shot steadying
- Robust/fast mobile computing important but challenging
Hardware limitations

- Most embedded devices do not have floating-point operations:
  - Power and heat
  - Speed
  - Precision

- Integer algorithms
  - Use less power because math units less complex
  - Execute faster than floating-point
  - Emulating floats is at least 10x slower
  - Enable added precision - can make every bit count
Integer Gaussian kernels

- Gaussian kernels are often used in machine learning/image/signal processing
  - Anomaly detection - non-outlier (normal) sample distribution
  - Support Vector Machines (SVM) kernels
  - Image smoothing
- Gaussian has several desirable characteristics:
  - Separable
  - Differentiable
  - Result of the central limit theorem
- Discrete Gaussian kernels should have integer values on embedded devices
Motivation

This work is motivated by:

1. The Gaussian kernel is the most ubiquitous kernel.
2. A discrete Gaussian kernel should exactly discretize a continuous Gaussian while consisting only of integers.
3. Using integers that are all powers-of-two enables bitwise operations and special processing in hardware.
The challenge

Here is a puzzler:

- Is it possible to design an integer kernel of arbitrary size whose entries fit a Gaussian function?
The challenge

Here is a puzzler:

- Is it possible to design an integer kernel of arbitrary size whose entries fit a Gaussian function?
- Is it possible to ensure that all the entries are powers-of-two?
The challenge

Here is a puzzler:

- Is it possible to design an integer kernel of arbitrary size whose entries fit a Gaussian function?
- Is it possible to ensure that all the entries are powers-of-two?
- Is there a particular sigma (a magic sigma) that meets both of these conditions?
The obligatory Gaussian equation slide

$$G(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$
Example: is this a Gaussian kernel and what is its $\sigma$?

Yes, it is a Gaussian kernel. Its $\sigma = \sqrt{\ln 4} \approx 0.85$. It is a magic kernel because all of its entries are powers-of-two!
Example: is this a Gaussian kernel and what is its $\sigma$?

Yes, it is a Gaussian kernel.

\[
\begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\]
Example: is this a Gaussian kernel and what is its $\sigma$?

Yes, it is a Gaussian kernel.

Its $\sigma = \frac{1}{\sqrt{\ln 4}} \approx 0.85$
Example: is this a Gaussian kernel and what is its $\sigma$?

Yes, it is a Gaussian kernel.

Its $\sigma = \frac{1}{\sqrt{\ln 4}} \approx 0.85$

It is a magic kernel because all of its entries are powers-of-two!
That sounds nice, but we must prove it!

\[
\begin{array}{ccc}
\sqrt{2} & 1 & \sqrt{2} \\
1 & 0 & 1 \\
\sqrt{2} & 1 & \sqrt{2}
\end{array}
\quad
\begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1
\end{array}
\]

Kernel distances \quad Kernel values

\[
G(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{x^2}{2\sigma^2} \right), \text{ with } \sigma = \frac{1}{\sqrt{\ln 4}}
\]
That sounds nice, but we must prove it!

<table>
<thead>
<tr>
<th>√2</th>
<th>1</th>
<th>√2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>√2</td>
<td>1</td>
<td>√2</td>
</tr>
</tbody>
</table>

Kernel distances

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Kernel values

- \( G(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \), with \( \sigma = \frac{1}{\sqrt{\ln 4}} \)
- \( G(0; \sigma) = \sqrt{\frac{\ln(4)}{2\pi}} \exp(0) = \sqrt{\frac{\ln(4)}{2\pi}} = C \)
That sounds nice, but we must prove it!

\[
\begin{array}{ccc}
\sqrt{2} & 1 & \sqrt{2} \\
1 & 0 & 1 \\
\sqrt{2} & 1 & \sqrt{2} \\
\end{array}
\quad
\begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\]

Kernel distances \quad Kernel values

- \( G(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{x^2}{2\sigma^2} \right) \), with \( \sigma = \frac{1}{\sqrt{\ln 4}} \)
- \( G(0; \sigma) = \sqrt{\frac{\ln(4)}{2\pi}} \exp(0) = \sqrt{\frac{\ln(4)}{2\pi}} = C \)
- \( G(1; \sigma) = C \exp \left( -\frac{\ln(4)}{2} \right) = C \exp(\ln(1/2)) = C/2 \)
That sounds nice, but we must prove it!

<table>
<thead>
<tr>
<th>$\sqrt{2}$</th>
<th>1</th>
<th>$\sqrt{2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$\sqrt{2}$</td>
<td>1</td>
<td>$\sqrt{2}$</td>
</tr>
</tbody>
</table>

Kernel distances

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Kernel values

- $G(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$, with $\sigma = \frac{1}{\sqrt{\ln 4}}$
- $G(0; \sigma) = \sqrt{\frac{\ln(4)}{2\pi}} \exp(0) = \sqrt{\frac{\ln(4)}{2\pi}} = C$
- $G(1; \sigma) = C \exp\left(-\frac{\ln(4)}{2}\right) = C \exp(\ln(1/2)) = C/2$
- $G(\sqrt{2}; \sigma) = C \exp\left(-\frac{2^2 \ln(4)}{2}\right) = C \exp(\ln(1/4)) = C/4$
That sounds nice, but we must prove it!

<table>
<thead>
<tr>
<th>Kernel distances</th>
<th>Kernel values</th>
<th>Scaled kernel values</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sqrt{2} )</td>
<td>1</td>
<td>( \sqrt{2} )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( \sqrt{2} )</td>
<td>1</td>
<td>( \sqrt{2} )</td>
</tr>
</tbody>
</table>

Kernel distances | Kernel values | Scaled kernel values

\[
G(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right), \text{ with } \sigma = \frac{1}{\sqrt{\ln 4}}
\]

\[
G(0; \sigma) = \sqrt{\frac{\ln(4)}{2\pi}} \exp(0) = \sqrt{\frac{\ln(4)}{2\pi}} = C
\]

\[
G(1; \sigma) = C \exp\left(-\frac{\ln(4)}{2}\right) = C \exp(\ln(1/2)) = C/2
\]

\[
G(\sqrt{2}; \sigma) = C \exp\left(-\frac{\sqrt{2}^2 \ln(4)}{2}\right) = C \exp(\ln(1/4)) = C/4
\]
Okay fine, but...

This is all well and good, but you **told** me the sigma. Maybe you just got lucky.

- By the way, how did you find this $\sigma$?
- Does this same $\sigma$ hold for any size kernel?
- Are there more general $\sigma$ values that result in Gaussian kernels that do not all have powers-of-two?
Contributions

1. We show how to combine integer arithmetic and bitwise shifts with integer Gaussian kernels.
2. We prove that there exists a set of “magic sigmas” that yield integer kernels whose entries are all powers-of-two.
3. We prove that the smallest sum “magic sigma” is $\frac{1}{\sqrt{\ln(4)}}$.
4. To enable flexible sigma values, we designed an integer kernels algorithm for any arbitrary sigma and kernel size.
Previous methods

Three main common approaches to integer kernel generation:

- Rounding method
- Binomial method
- Convolution method

Limitations:

- Result in approximations of the Gaussian kernel
- Do not necessarily result in integer values that are powers-of-two

Thus, a more systematic analysis of the kernel construction is necessary in order to derive exact integer kernels
Method 1: Rounding method

1. Create a floating-point Gaussian kernel
2. Normalize the values to make its smallest entry 1
3. Round the entries and store them as integers
4. Divide the kernel by the kernel sum

\[ \sigma = 1, \quad r = 2\sigma \]

\[ \sigma = 10, \quad r = 2\sigma \]
Method 2: Binomial method

- Use odd-length rows of Pascal’s triangle: [1 2 1], [1 4 6 4 1]
- Does not enable control over $\sigma$ for a particular kernel size
- The $\sigma$ increases with increasing kernel size

\[
\begin{array}{c}
\sigma = 1.0565 \\
\sigma = 1.3723
\end{array}
\]
Method 3: Convolution method

- Convolve smaller integer kernels with each other
- Enables the generation of a range of $\sigma$ values depending on the number of convolutions
- Based on the relationships
  - $G_\sigma = G_{\sigma_1} \ast G_{\sigma_2}$
  - $\sigma^2 = \sigma_1^2 + \sigma_2^2$.
- Approximation errors increase with larger kernels because of convolution of truncated smaller kernels
- When kernels are chosen from the odd rows of Pascal’s triangle, the resulting kernel will be another odd row in Pascal’s triangle
Power-of-two Gaussian kernels require the following constraints:

1. Consistent sigma
2. Integer kernel values
3. Kernel values are powers-of-two

We will demonstrate how to ensure all of these conditions
Notation

- $x_j$: the distance from the center to a location in the kernel
- $x_m$: the farthest distance from the center in the kernel
- $G(x_j)$: the value of the Gaussian evaluated at $x_j$
- $R_{x_i,x_j} = \frac{G(x_i)}{G(x_j)}$: the ratio of the Gaussian values at two locations in the kernel
- $\sigma_{x_i,x_j}$: the Gaussian $\sigma$ found using points $x_i$ and $x_j$
What is meant by “consistent” $\sigma$?

- Given a desired $\sigma$, the relationship of every pair of points should conform to this value.
- Since Gaussian functions vary by distance, we look at the distances in a kernel.

<table>
<thead>
<tr>
<th></th>
<th>$\sqrt{2}$</th>
<th>1</th>
<th>$\sqrt{2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sqrt{2}$</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$\sqrt{2}$</td>
<td>1</td>
<td>$\sqrt{2}$</td>
<td></td>
</tr>
</tbody>
</table>

Distances for $3 \times 3$ kernel.

<table>
<thead>
<tr>
<th></th>
<th>$2\sqrt{2}$</th>
<th>$\sqrt{5}$</th>
<th>2</th>
<th>$\sqrt{5}$</th>
<th>$2\sqrt{2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sqrt{5}$</td>
<td>$\sqrt{2}$</td>
<td>1</td>
<td>$\sqrt{2}$</td>
<td>$\sqrt{5}$</td>
<td></td>
</tr>
<tr>
<td>$\sqrt{2}$</td>
<td>1</td>
<td>0</td>
<td>$\sqrt{2}$</td>
<td>$\sqrt{5}$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$\sqrt{5}$</td>
<td>$\sqrt{2}$</td>
<td>1</td>
<td>$\sqrt{2}$</td>
<td>$\sqrt{5}$</td>
</tr>
<tr>
<td>$2\sqrt{2}$</td>
<td>$\sqrt{5}$</td>
<td>2</td>
<td>$\sqrt{5}$</td>
<td>$2\sqrt{2}$</td>
<td></td>
</tr>
</tbody>
</table>

Distances for $5 \times 5$ kernel.
Ratios of Gaussians

Any two points $x_i, x_j$ are needed to define the $\sigma$ of a Gaussian.

$$G(x; \sigma) = \frac{1}{\sqrt{2\pi \sigma}} \exp \left( -\frac{x^2}{2\sigma^2} \right)$$

Taking their ratio makes the points scale invariant.

$$R_{x_i, x_j} = \frac{G(x_i)}{G(x_j)} = \frac{\exp \left( -\frac{x_i^2}{2\sigma^2} \right)}{\exp \left( -\frac{x_j^2}{2\sigma^2} \right)} = \exp \left( \frac{x_j^2 - x_i^2}{2\sigma^2} \right)$$

Solving for $\sigma$ leads to

$$\sigma_{x_i, x_j} = \sqrt{\frac{x_j^2 - x_i^2}{2 \ln(R_{x_i, x_j})}}$$
Solving for $\sigma$

We can simplify $R_{x_i,x_j}$ by using $G(0)$ as a reference point.

$$R_{0,x_j} = \frac{G(0)}{G(x_j)} = \exp \left( \frac{x_j^2}{2\sigma^2} \right)$$

$$R_{0,1} = \frac{G(0)}{G(1)} = \exp \left( \frac{1}{2\sigma^2} \right)$$

Using $R_{0,x_j}$, we can solve for $\sigma_{0,x_j}$

$$\sigma_{0,x_j} = \sqrt{\frac{x_j^2}{2 \ln(R_{0,x_j})}}$$
**Solving for $\sigma$**

We can simplify $R_{x_i,x_j}$ by using $G(0)$ as a reference point.

\[
R_{0,x_j} = \frac{G(0)}{G(x_j)} = \exp \left( \frac{x_j^2}{2\sigma^2} \right)
\]

\[
R_{0,1} = \frac{G(0)}{G(1)} = \exp \left( \frac{1}{2\sigma^2} \right)
\]

Using $R_{0,x_j}$, we can solve for $\sigma_{0,x_j}$

**Consistent $\sigma$**

\[
\sigma_{0,x_j} = \sqrt{\frac{x_j^2}{2 \ln(R_{0,x_j})}} = \sqrt{\frac{x_j^2}{2 \ln \left( (R_{0,1})^{x_j^2} \right)}}
\]
Solving for $\sigma$

We can simplify $R_{x_i,x_j}$ by using $G(0)$ as a reference point.

$$R_{0,x_j} = \frac{G(0)}{G(x_j)} = \exp \left( \frac{x_j^2}{2\sigma^2} \right)$$

$$R_{0,1} = \frac{G(0)}{G(1)} = \exp \left( \frac{1}{2\sigma^2} \right)$$

Using $R_{0,x_j}$, we can solve for $\sigma_{0,x_j}$

Consistent $\sigma$

$$\sigma_{0,x_j} = \sqrt{\frac{x_j^2}{2\ln(R_{0,x_j})}} = \sqrt{\frac{x_j^2}{2\ln((R_{0,1})^{x_j^2})}} = \sqrt{\frac{1}{2\ln(R_{0,1})}}$$

Thus, when $R_{0,x_j} = (R_{0,1})^{x_j^2}$ for all $x_j$, $\sigma_{0,x_j}$ is “consistent” because it does not depend on $x_j$. 
Ensuring Integer Values

Next, we need to ensure that all values are integers.

- We can find the value at any location $x_j$ in the kernel $G(x_j)$ relative to the value at the center $G(0)$.

\[ G(x_j) = \frac{G(0)}{R_{0,x_j}} = \frac{G(0)}{(R_{0,1})^{x_j^2}} \]

- By definition, the smallest value in the kernel $G(x_m)$ will be at the farthest distance $x_m$ from the center.

**Ensuring integer values**

If $R_{0,1} = \frac{p}{q}$ is rational and $G(0) = (p)^{x_m^2}$, all values in the kernel will be integers.

- This can easily be proven (see proof in paper).
Ensuring powers-of-two: the “magic sigma” family

- Powers-of-two enable efficient bit-wise shifts
- Recall that $R_{0,x_j} = (R_{0,1})^{x_j^2}$
- If $R_{0,1}$ is a power-of-two, then all kernel entries are powers-of-two because $x_j^2$ is always an integer.
- Thus, $R_{0,1} = 2^i$ (where $i > 0$ is an integer) leads to the “magic sigmas”
Ensuring powers-of-two: the “magic sigma” family

- Powers-of-two enable efficient bit-wise shifts
- Recall that $R_{0,x_j} = (R_{0,1})^{x_j^2}$
- If $R_{0,1}$ is a power-of-two, then all kernel entries are powers-of-two because $x_j^2$ is always an integer.
- Thus, $R_{0,1} = 2^i$ (where $i > 0$ is an integer) leads to the “magic sigmas”

The “magic sigma” family

$$
\sigma = \frac{1}{\sqrt{2 \ln(2^i)}}
$$
The ultimate “magic sigma”

We can go one step further by setting $R_{0,1} = 2$

- This yields the smallest kernel values (see proof in paper)
- This also yields the largest $\sigma$ for a power-of-two Gaussian kernel (see proof in paper)
We can go one step further by setting $R_{0,1} = 2$

- This yields the smallest kernel values (see proof in paper)
- This also yields the largest $\sigma$ for a power-of-two Gaussian kernel (see proof in paper)
Ingredients needed to ensure power-of-two Gaussian kernels:

- Consistent $\sigma$: $\sigma_{0,x_j} = \frac{1}{\sqrt{2 \ln(R_{0,1})}}$
- Integer constraint: $R_{0,1}$ must be rational.
- Power-of-two: $R_{0,1} = 2^i$, where $i > 0$ is an integer

This leads to the “magic sigma” when $i = 1$:

$$\sigma = \frac{1}{\sqrt{\ln(4)}} \approx 0.849$$
Enabling greater $\sigma$ flexibility

Magic sigmas

- Advantage: efficient computation via bitwise shifts
- Disadvantage: limited $\sigma$ options (max $\approx 0.849$)
- Tradeoff: power-of-two kernels vs. flexible $\sigma$
- Need algorithm for integer Gaussian kernels for any specified $\sigma$
**General algorithm for integer Gaussian kernels**

Input: Gaussian $\sigma$, kernel radius $r$, and kernel dimension $D$

Output: Integer Gaussian kernel

1. Calculate real number $R'_{0,1} = \exp\left(\frac{1}{2\sigma^2}\right)$.
2. Since $R_{0,1}$ must be rational, find the “best rational approximation” $R_{0,1} = \frac{p}{q}$ of integers that approximates $R'_{0,1}$ using the “continued fraction”
3. Calculate the Euclidean distance from the kernel center to each kernel entry $x_j$, where $j = 1\ldots(2r + 1)^D$
4. Calculate $G(0) = (p)^{x_m^2}$
5. Find each kernel entry $G(x_j) = \frac{G(0)}{(R_{0,1})^{x_j}}$
6. Apply the kernel to the image and divide the result by the kernel sum
Example: $3 \times 3$ kernel using “magic sigma” $\sigma = \frac{1}{\sqrt{\ln(4)}}$

1. Calculate $R'_{0,1} = \exp \left( \frac{1}{2\sigma^2} \right) = \exp \left( \frac{\ln(4)}{2} \right) = 2$

2. Find the “best rational approximation” $R_{0,1} = \frac{p}{q} = \frac{2}{1}$

3. Calculate the Euclidean distance from the kernel center to each kernel entry $x_j = 0, 1, \sqrt{2} \rightarrow x_m = \sqrt{2}$

4. Calculate $G(0) = (p)^{x_m^2} = (2)^{\sqrt{2}^2} = 4$

5. Find each kernel entry $G(x_j) = \frac{G(0)}{(R_{0,1})^{x_j^2}}$

   $G(1) = \frac{4}{2^{(1)^2}} = 2$ and $G(\sqrt{2}) = \frac{4}{(2)(\sqrt{2})^2} = 1$

6. Apply the kernel to the image and divide the result by the kernel sum
### 3 × 3 Gaussian Integer Kernels with $R_{0,1}$ Set to an Integer

<table>
<thead>
<tr>
<th>$R_{0,1} = 2$</th>
<th>$R_{0,1} = 3$</th>
<th>$R_{0,1} = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma \approx 0.85$</td>
<td>$\sigma \approx 0.67$</td>
<td>$\sigma \approx 0.60$</td>
</tr>
<tr>
<td>1 2 1</td>
<td>1 3 1</td>
<td>1 4 1</td>
</tr>
<tr>
<td>2 4 2</td>
<td>3 9 3</td>
<td>4 16 4</td>
</tr>
<tr>
<td>1 2 1</td>
<td>1 3 1</td>
<td>1 4 1</td>
</tr>
</tbody>
</table>

1 2 1
### 3 × 3 Gaussian integer kernels with \( R_{0,1} \) a rational number

<table>
<thead>
<tr>
<th>( R_{0,1} )</th>
<th>Kernel</th>
<th>( \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 3/2 )</td>
<td>4 6 4</td>
<td>( 1.11 )</td>
</tr>
<tr>
<td>( 5/4 )</td>
<td>16 20 16</td>
<td>( 1.50 )</td>
</tr>
<tr>
<td>( 11/10 )</td>
<td>100 110 100</td>
<td>( 2.29 )</td>
</tr>
</tbody>
</table>

\[ R_{0,1} = \frac{3}{2}, \quad \sigma \approx 1.11 \]

\[ R_{0,1} = \frac{5}{4}, \quad \sigma \approx 1.50 \]

\[ R_{0,1} = \frac{11}{10}, \quad \sigma \approx 2.29 \]
$5 \times 5$ Gaussian “magic sigma” integer kernel

### Gaussian kernel values

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>8</th>
<th>16</th>
<th>8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>64</td>
<td>128</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>128</td>
<td>16</td>
</tr>
<tr>
<td>16</td>
<td>128</td>
<td>256</td>
<td>128</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>128</td>
<td>64</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>16</td>
<td>8</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

$R_{0,1} = 2, \sigma \approx 0.849$
$5 \times 5$ Gaussian “magic sigma” integer kernel

<table>
<thead>
<tr>
<th>Gaussian kernel values</th>
<th>Power-of-two exponents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 8 16 8 1</td>
<td>0 3 4 3 0</td>
</tr>
<tr>
<td>8 64 128 64 8</td>
<td>3 6 7 6 3</td>
</tr>
<tr>
<td>16 128 256 128 16</td>
<td>4 7 8 7 4</td>
</tr>
<tr>
<td>8 64 128 64 8</td>
<td>3 6 7 6 3</td>
</tr>
<tr>
<td>1 8 16 8 1</td>
<td>0 3 4 3 0</td>
</tr>
</tbody>
</table>

$R_{0,1} = 2, \sigma \approx 0.849$

The kernel values can become large for large kernels. Solution: store the exponents, and use them for bitwise shifting
5 × 5 Gaussian integer kernels with $R_{0,1}$ a rational number

$$R_{0,1} = \frac{3}{2}, \sigma \approx 1.1105$$

- The kernel values increase with increasing numerator and denominator of $R_{0,1}$ and as the kernel size increases
- Separable 1D filters can instead be used
- If kernel becomes large, Fourier domain can be used
Exact integer results compared with the rounding approach

Exact Integers Compared with Rounding Method ($\sigma = 1$)

$\sigma = 1$, $R_{0,1} = \frac{61}{37}$

Exact Integers Compared with Rounding Method ($\sigma = 5$)

$\sigma = 5$, $R_{0,1} = \frac{101}{99}$
Exact integer results compared with the basic rounding approach

$\sigma = 7$, $R_{0,1} = \frac{98}{97}$

$\sigma = 10$, $R_{0,1} = \frac{200}{199}$
Summary: contributions

1. We proved for the first time the existence of a set of “magic sigmas” that generate integer Gaussian kernels whose entries are powers-of-two.

2. The magic sigma values are \( \sigma = \frac{1}{\sqrt{2 \ln(2^i)}} \).

3. The simplest and largest of these \( \sigma = \frac{1}{\sqrt{\ln(4)}} \approx 0.849 \).

4. We developed a flexible algorithm for generating integer Gaussian kernels for arbitrary \( \sigma \) and kernel size.
Discussion points: what does this all mean?

- Integer operations improve power, speed, and precision on mobile devices
- If possible, choose a “magic sigma” rather than an arbitrary number. This enables advantage of integer kernels and bit-wise shifts
- For flexible sigma, use the algorithm to generate integer kernels
- Efficient algorithm implementations can take advantage of these kernels