Cumulative Gaussian Curve Fitter for Boundary Parameterization

David Wang[1, 2], Robert Tamburo[3], George Stetten[1, 2, 3]

[1] Department of Biomedical Engineering, Carnegie Mellon University, Pittsburgh PA 15213, USA
[2] University of Pittsburgh Medical Center, Pittsburgh PA 15261, USA
[3] Department of Bioengineering, University of Pittsburgh, Pittsburgh, PA 15261, USA

Abstract. We have previously developed an algorithm for locating boundaries in an image with sub-pixel resolution, as well as estimating boundary width and image intensity within the adjoining objects. The algorithm operates by finding the parameters of a cumulative Gaussian curve that best approximates an intensity profile taken across a boundary. If intensity is sampled along the image gradient across a boundary, it is reasonable to assume the profile approximates a finite portion of a cumulative Gaussian. Given that assumption, the first derivative of the profile should be the corresponding portion of a Gaussian, completely described by its mean, standard deviation, and amplitude. We present here a simple and rapid method to find those parameters, given that we only have a potentially skewed sample of the Gaussian. The parameters are approximated first for the finite sample, and then both ends of the Gaussian are extrapolated using the resulting parameters. New parameters are then calculated and the procedure is repeated. The optimization rapidly converges, yielding boundary location (mean) with sub-pixel accuracy as well boundary width (standard deviation). Integration then reproduces the cumulative Gaussian, and a least-squares fit is applied to estimate the constant of integration, from which intensity of the adjoining regions can be estimated.

1 Introduction

Medical images rarely contain sharp boundaries between objects, and determining where one object ends and another begins is not a trivial task. Pixel intensities sampled normal to the boundary tend to assume the shape of a cumulative Gaussian function, in which the intensity of one object gradually shifts to that of the other, even if the actual anatomical boundaries are sub-pixel in resolution. The resulting step function is convolved with the impulse response of the imaging system. We have conducted previous work in this area [1], in which we found a profile of the boundary by forming an ellipse whose major axis lies along a boundary gradient and then by projecting voxels within that ellipse onto the major axis. We explored two different curve-fitters to fit a cumulative Gaussian to this profile, neither of which were tailored to the cumulative Gaussian, and which were therefore unnecessarily slow and otherwise burdensome. The curve-fitter we present here is tailored specifically to the cumulative Gaussian and uses its known statistics to rapidly converge.

We start by reviewing the cumulative Gaussian function in Section 2. We then present our method in Section 3, our API in Section 4, experimental results in Section 5, and conclusions in Section 6.

2 The Cumulative Gaussian Function

A number of different functions could potentially be fit to an intensity profile across a boundary. We choose the cumulative Gaussian for the following reason: Most anatomical boundaries are basically step functions at a sub-millimeter scale, where one tissue ends and another begins. Image acquisition is always limited in resolution, with each imaging device exhibiting a point spread function, usually at a significantly larger scale than the actual tissue boundary. Additional blurring may be performed intentionally or inevitably during image reconstruction and processing. These sequential convolutions tend to have the net effect of convolving with a Gaussian kernel because of the Central Limit Theorem. Thus we can expect the step function of the anatomical boundary to reach the image analysis stage as a cumulative Gaussian, i.e., the convolution of a step function with a Gaussian. By fitting a cumulative Gaussian, we may, in effect, reverse the blurring and parameterize the original boundary.

Fitting a cumulative Gaussian to a boundary profile requires optimizing four parameters: (1) standard deviation, corresponding to the boundary width, (2) mean, corresponding to boundary location, and (3,4) two asymptotic values corresponding to voxel intensity on either side of the boundary far enough from the boundary to be unaffected by blurring.

The cumulative Gaussian is derived as follows: The normalized Gaussian,

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} , \tag{1}$$

is integrated to yield the error function (*erf*),

$$\int_{-x}^{x} G(v)dv = erf\left(\frac{x-\mu}{\sigma\sqrt{2}}\right) , \tag{2}$$

which is scaled and offset to yield the cumulative Gaussian $C(x)$ as follows:

$$C(x) = I_1 + \frac{I_2 - I_1}{2}\left(1 + erf\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right) . \tag{3}$$

The four fixed parameters of $C(x)$ are μ (mean), σ (standard deviation), and I_1 and I_2 (asymptotic voxel intensity inside and out). On one side of the boundary

$$erf(-\infty) = -1,\ C(-\infty) = I_1 \tag{4}$$

and on the other side

$$erf(\infty) = 1,\ C(\infty) = I_2. \tag{5}$$

The four parameters are labeled in Fig. 1, which shows a particular fit of the cumulative Gaussian along the sampled portion of the boundary profile.

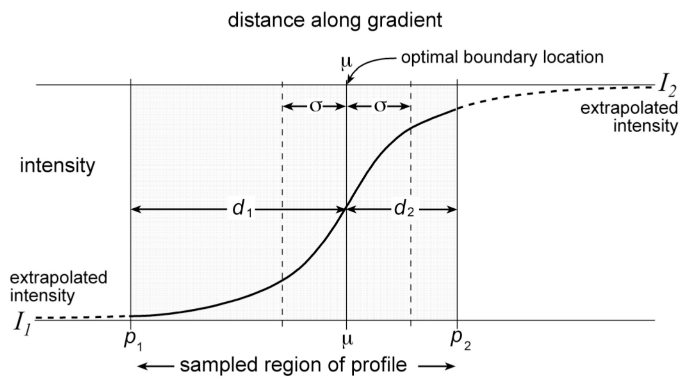

Fig. 1 A cumulative Gaussian fit to a profile from a sampled region (between p_1 and p_2).

A variety of techniques is available for optimizing the fit of a non-linear function to a set of sample points. The ITK library includes Levenberg-Marquardt (LM) optimization, for example. In our previous work with ellipsoids, we used a Quasi-Newton non-linear optimization algorithm, provided as part of the *AD Model Builder* from Otter Research, Inc., [2] because unlike LM it does not require explicit derivatives and is acceptably rapid and robust.

Assuming that the function to which we want to fit is the cumulative Gaussian, we can use a curve fitter tailored specifically to that function, which reduces computation, as is described in the following section. Furthermore, unlike the two curve-fitters above, ours does not require parameter initialization.

3 Method

To convert a cumulative Gaussian into a Gaussian requires differentiation, which in the discrete domain may be accomplished in a variety of ways. We chose the following *difference function*: Given $s[n]$ sampled from the continuous function $s(x)$ for samples $n = 0,1,2,\ldots,N$ such that $s[n] = s(x_0 + n(\Delta x))$, we define the difference function $s'[n]$ on one fewer samples, $n = 1,2,...,N$ such that $s'[n] = s[n] - s[n-1]$, with the understanding that the location of $s'[n]$ is $x_0 + (n-1/2)(\Delta x)$, half way between sample $s[n]$ and $s[n-1]$. This avoids sampling bias along the x dimension, while preserving maximal resolution, by using only two samples to compute the difference at each point. (Δx is the distance between samples and x_0 is considered the origin).

We aim to fit a Gaussian to $s'[n]$ to minimize the error

$$E(\mu,\sigma,a) = \sum_{n=1}^{N} \left(G[n] - s'[n]\right)^2 \tag{6}$$

where $G[n]$ is the discrete version of the Gaussian $G(x)$ defined in Equation 1. We may estimate the three parameters, μ, σ^2, and A, by treating $s'[n]$ as though it were the histogram of a population. Pretend, if you will, that n represents the number of *hats* that a person owns, and $s'[n]$ is the number of people in a population who each own n hats, i.e., $s'[n]$ a histogram. The average number of hats per person is then

$$\mu = \frac{\text{total number of hats}}{\text{total number of people}} = \frac{\sum_{n=1}^{N} n \cdot s'[n]}{\sum_{n=1}^{N} s'[n]}, \tag{7}$$

and likewise, assuming a Gaussian distribution, the other two parameters are

$$\sigma^2 = \frac{\sum_{n=1}^{N} (n-\mu)^2 \cdot s'[n]}{\sum_{n=1}^{N} s'[n] - 1} \quad \text{and} \quad A = \frac{\sum_{n=1}^{N} s'[n]}{\sqrt{2\pi\sigma}}. \tag{8}$$

These parameters fully specify a Gaussian function, which we can construct, not just in the sampled region, but for all n extending in both directions from the sampled data. We extend $s'[n]$ with $G[n]$ in both directions over a total of $3N$ samples and recompute μ, σ^2, and A. Repeating these steps quickly converges (see results below) with each step being computationally very simple.

What we really want, however, is not the Gaussian $G[n]$ that optimally fits $s'[n]$, but rather the cumulative Gaussian $C[n]$ that optimally fits the original boundary profile $s[n]$. For this we need to solve for a fourth variable, inherent in the relationship between $s'(x)$ and $s(x)$, namely, the constant of integration c in

$$s(x) = \int_{-\infty}^{+\infty} s'(x)dx + c. \tag{9}$$

We want to minimize the following error function:

$$E(c) = \sum_{n=0}^{N} \left(C[n] + c - s[n]\right)^2, \tag{10}$$

which we accomplish by taking its derivative with respect to c and equating it to 0,

$$\frac{d\sum_{n=0}^{N}\left(C[n]+c-s[n]\right)^2}{dc}=2\sum_{n=0}^{N}\left(C[n]+c-s[n]\right)=0. \tag{11}$$

We can then solve for c,

$$c=\frac{\sum_{n=0}^{N}C[n]-s[n]}{N}. \tag{12}$$

4 API within ITK

The cumulative Gaussian optimizer was contributed to ITK in July of 2003. It can be found in the toolkit as *CumulativeGaussianOptimizer* in the *Numerics* subdirectory. The optimizer was implemented as a *MultipleValuedNonLinearOptimizer* since it optimizes a multiple valued function, returning multiple values when given a set of parameters. Unlike other optimizers in ITK that accept arbitrary cost functions, *CumulativeGaussianOptimizer* specifically uses the *CumulativeGaussianCostFunction*. The cost function needs to be instantiated and initialized by setting the number of data samples and a pointer to the data. The data is passed to the cost function so that a fit error can be calculated after the optimization is complete.

Once the optimizer is instantiated, it needs to know the sampled data points to which a cumulative Gaussian curve is to be fitted. These data points are initialized via *SetDataArray*. The *DifferenceTolerance* variable determines when the optimizer should stop. The default value is 1×10^{-10}. The variable *Verbose* determines whether computational details are displayed. The default value is *false*.

The optimization starts when *StartOptimization* is invoked. Once the optimization is complete, the parameters of the fitted cumulative Gaussian can be retrieved via Get Macros defined for *ComputedMean*, *ComputedStandardDeviation*, *UpperAsymptote*, *LowerAsymptote*, *FitError*, and *FinalSampledArray*.

5 Experimental Results

A set of 3 cumulative Gaussian test curves each containing 10 samples was generated with varying degrees of skew ($\mu = 2,\ 3,\ 5,\ \sigma = 2,\ I_1 = 0,\ I_2 = 100$). As shown in Figure 2, when there is minimal skew ($\mu = 5$ close to the middle of the 10 samples) the curve fitter converged in less than 5 iterations, whereas with greater skew ($\mu = 2,\ 3$) greater numbers of iterations were required. Each iteration was, however, very fast, involving a limited number of simple arithmetic operations.

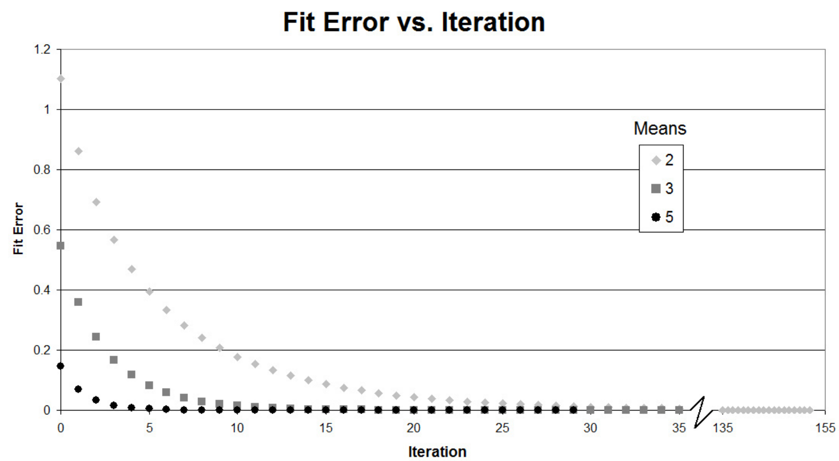

Fig. 2 RMS Error fitting test curves with various means to the cumulative Gaussian, as a function of iteration number.

6 Conclusions

We have developed a rapid curve fitter specifically for the cumulative Gaussian and implemented it in ITK. Current work is continuing to develop uses for this method in automated segmentation, based on profiles sampled across boundaries.

Acknowledgements

Funded by a contract with the National Library of Medicine. Thanks to Wendy Stetten for the hat analogy.

References

1) R. Tamburo, G. Stetten, “Gradient-Oriented Profiles for Automated Unsupervised Boundary Classification and their Application to Core Atoms Towards Shape Analysis,” *International Journal of Image and Graphics*, vol. 1, no. 4, pp. 659-680, 2001.
2) Otter Research, Ltd, AD Model Builder. http://otter-rsch.com/admodel.htm.