Gaussian Processes for Classification

Amir Atiya
Dept Computer Engineering, Cairo University
amir@alumni.caltech.edu
www.alumni.caltech.edu/~amir
Currently on leave at Veros Systems, TX

February 2011

Gaussian Process Classification

- Nonparametric classification method.

- Based on a **Bayesian** methodology. It assumes some prior distribution on the underlying probability densities that guarantees some smoothness properties.

- The final classification is then determined as the one that provides a good fit for the observed data, while at the same time guaranteeing smoothness.

- This is achieved by taking the smoothness prior into account, while factoring in the observed classification of the training data.

- It is a very effective classifier. We have recently performed a large scale comparison study of 12 major classifiers, on 22 benchmark classification problems. The Gaussian process classifier was the best classifier among all.

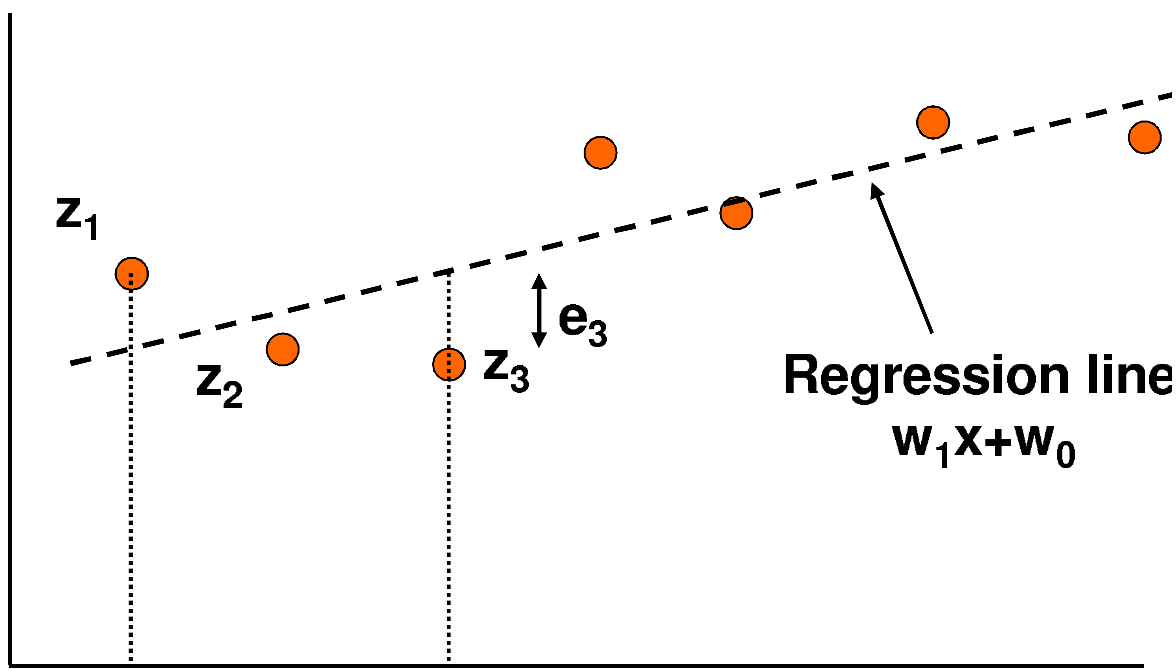
- It was developed in the geostatistics field in the seventies (O'Hagan and others).

- Was popularized in the machine learning community by MacKay, Williams and Rasmussen.

Overview of Bayesian Parameter Estimation

- Consider a model whose function depends on certain parameters.
- Assume a prior distribution for these parameters.
- Factor in the observed data, to obtain a posterior distribution of the parameters.
- Obtain a prediction for a new point, by estimating its distribution given that we know the posterior of the parameters.

Example: A linear regression problem:



Bayesian Parameter Estimation (Contd)

- The regression model is given by $z = w^T x$.

- Assume a prior for the parameters $p(w)$, e.g. zero mean Gaussian.

- Observe a number of points: $(x_i, z_i),\ i = 1, \ldots, N$ (let the data points be D).

- The posterior distribution of the parameters is given by:

$$p(w|D) = p(D|w)p(w)/p(D)$$

where

$$p(D|w) = \prod_i \frac{e^{-(z_i - w^T x_i)^2/(2\sigma^2)}}{\sqrt{2\pi}\sigma}$$

- Consider a new points x^*, at which we would like to predict the function z^*.

- Then

$$\begin{aligned} p(z^*|D) &= \int p(z^*, w|D) dw \\ &= \int p(z^*|w) p(w|D) dw \end{aligned}$$

On the Bayes Classifier

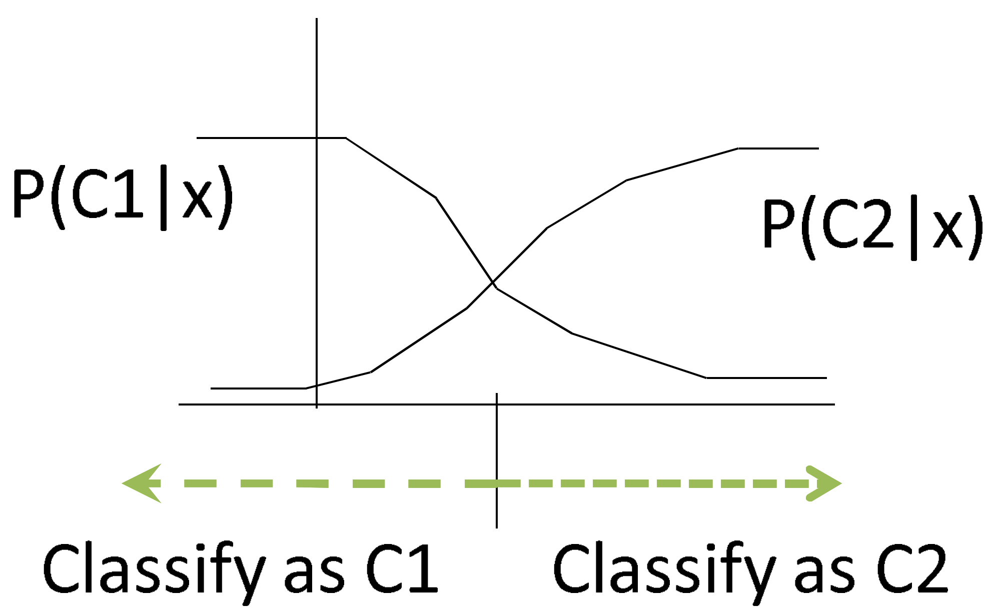
- **Class-conditional densities** $p(x|C_k)$, where x is the feature vector, C_k represents class k. This gives the probability density of feature vector x that is coming from class C_k.

- **Posterior probabilities** $P(C_k|x)$. It represents the probability that the pattern x comes from class C_k.

- By Bayes rule:

$$P(C_k|x) = \frac{p(x|C_k)P(C_k)}{p(x)}$$

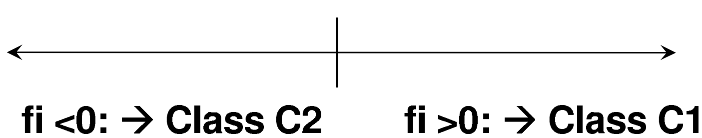
- Classify x on the basis of the value of $P(C_k|x)$. Select the class C_k giving maximum $P(C_k|x)$.



The Gaussian Process Classifier

- It focuses on modeling the **posterior probabilities**, by defining certain latent variables: f_i is the **latent variable** for pattern i.

- Consider a two-class case: f_i is a measure of the degree of membership of class C_1, meaning:
  - If f_i is positive and large $\longrightarrow$ pattern i belongs to class C_1 with large probability.
  - If f_i is negative and large in magnitude $\longrightarrow$ pattern i belongs to class C_2 with large probability.
  - If f_i is close to zero, class membership is less certain.

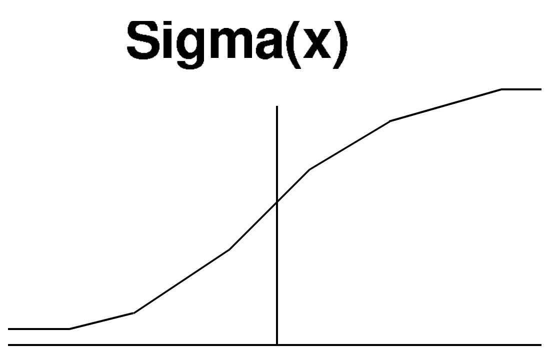


The Gaussian Process Classifier (Contd)

- Let $y_i = 1$ ($y_i = -1$) denote that pattern i belongs to class C_1 (C_2).

- The posterior probability (for class C_1) is:

$$
\begin{aligned}
P(C_1|x_i) &\equiv P(y_i = 1|f_i) \\
&= \sigma(f_i) \\
&\equiv \int_{-\infty}^{f_i} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx
\end{aligned}
$$



More Definitions

- Arrange the f_i's of the training set in a vector $f \equiv (f_1, \ldots, f_N)^T$.

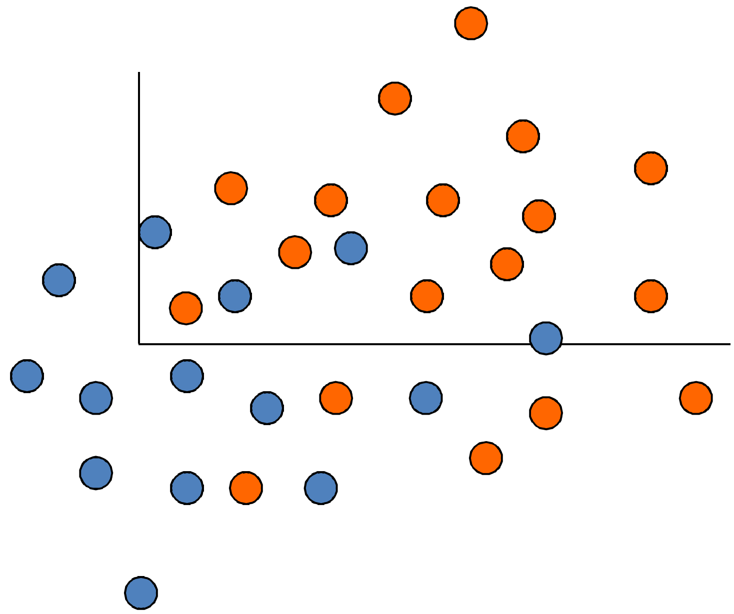
- Arrange the class memberships y_i of the training set in a vector $y \equiv (y_1, \ldots, y_N)^T$.

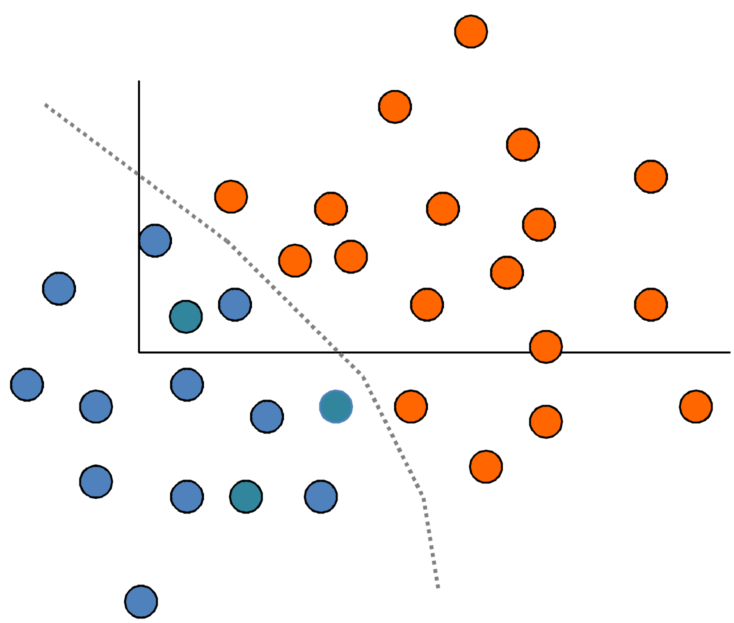
- Let x_i be the feature vector of training pattern i.

- Define the training matrix X as that containing all training vectors x_i.

- Let x_* be a testing vector to be classified, with latent variable f_* and class membership y_*.

Smoothness Prior





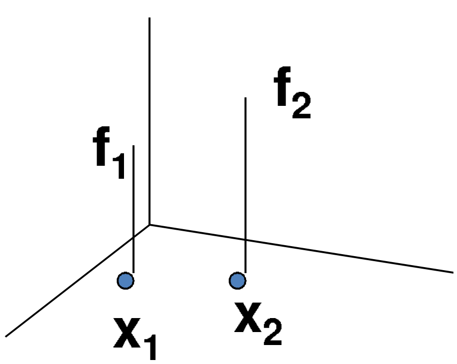
Smoothness Priors (Contd)

- We enforce smoothness by defining a prior on the latent variables f_i.

- Patterns with close by feature vectors x_i will have *highly correlated* latent variables f_i.

$$p(f|X) = \mathcal{N}(f, 0, \Sigma)$$

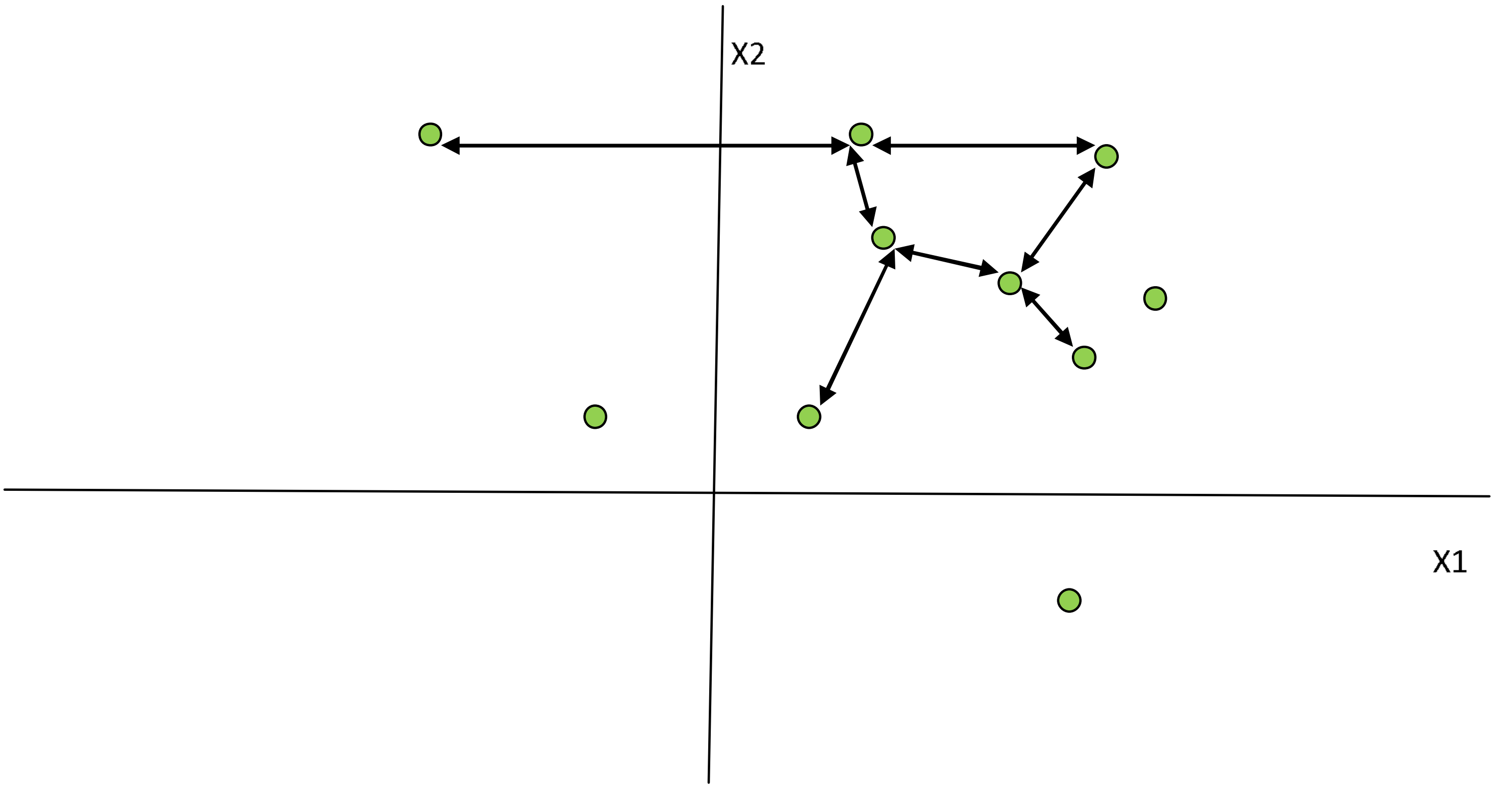
where $\mathcal{N}(f, \mu, \Sigma)$ denotes a Gaussian density of variable f having mean vector μ and covariance matrix Σ.



$Corr(f_1, f_2) = fn(||x_1 - x_2||)$

e.g. $\exp(-\alpha \, ||x_1 - x_2||^2)$

Smoothness Priors (Contd)



Classification

Consider a test pattern. Using standard probability manipulations, we get the probability that the test pattern belongs to class C_1:

$$J_* \equiv p(y_* = +1|X, y, x_*) = \int \sigma(f_*) p(f_*|X, y, x_*) df_*$$

(Recall that $\sigma(f_*) \equiv P(y_* = 1|f_*)$.)

$$p(f_*|X, y, x_*) = \int p(f_*|X, x_*, f) p(f|X, y) df$$

where

$$p(f|X, y) = \frac{p(y|f) p(f|X)}{p(y|X)}$$

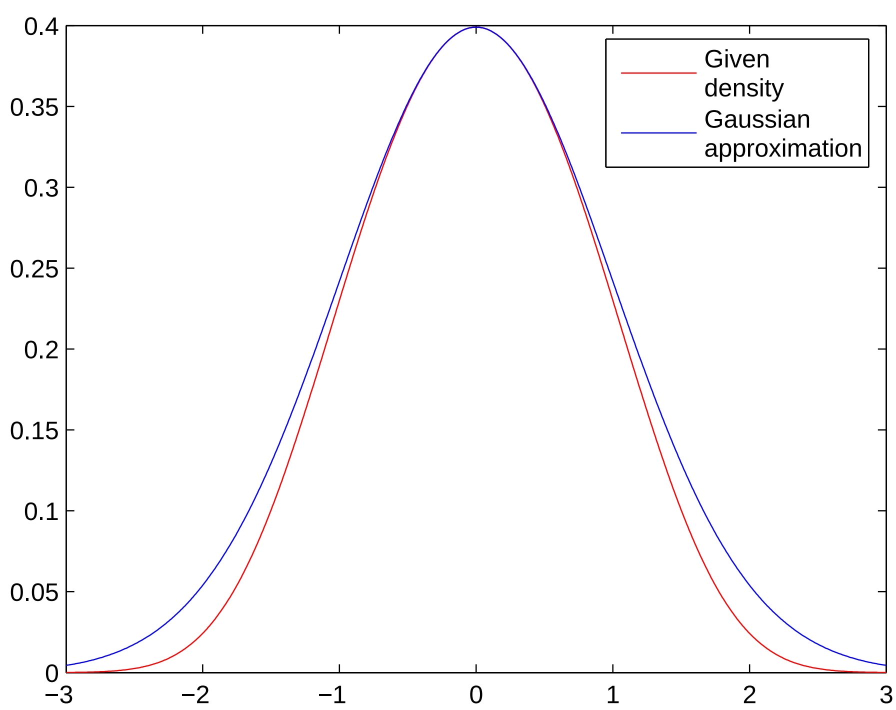
Classification (Contd)

- As we can see, to classify a point we have to evaluate an N-dimensional integral, where N is the size of the training set.

- This integral is intractable.

- There are some approximations, such as:
  - Laplace approximation,
  - Expectation propagation.

- Or, one can evaluate it using the Markov-Chain-Monte-Carlo (MCMC) procedure. This is numerically a very slow procedure.



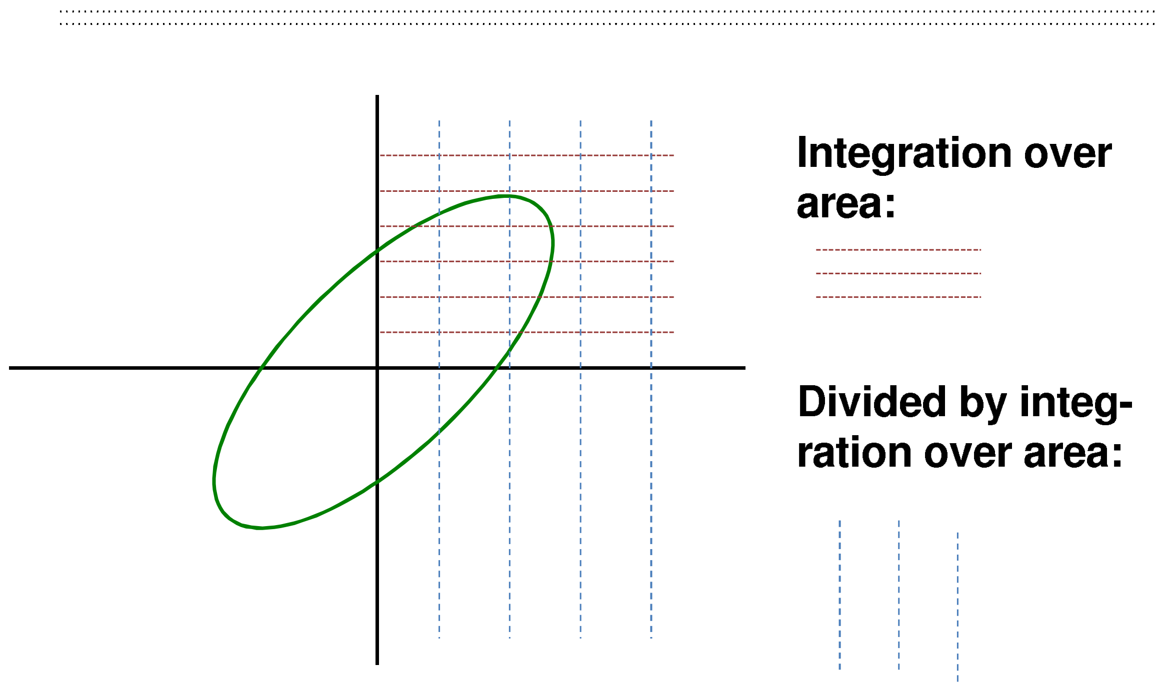
The Proposed Method

- We use several variable transformations.
- We also implement several matrix manipulations and simplifications.
- These result in the following formula for the classification of a test pattern:

$$J_* = p(y = 1|X, y, x_*) = \frac{\int_{orth} \mathcal{N}\left(v, 0, I + A_{12}\Sigma' A_{12}\right) dv}{\int_{orth+} \mathcal{N}\left(v, 0, I + A_{12}\Sigma' A_{12}\right) dv} \equiv \frac{I_1}{I_2}$$

where $v = (v_1, \ldots, v_{N+1})^T$, $orth$ means the orthant $v \geq 0$, $orth+$ means $-\infty < v_1 < \infty, \quad v_2 \geq 0, \ldots, v_{N+1} \geq 0$, $\mathcal{N}$ is the multivariate Gaussian density with covariance matrix $I + A_{12}\Sigma' A_{12}$, given by: $A_{12} = \begin{bmatrix} -1 & 0 \\ 0 & C' \end{bmatrix}$, $\Sigma' = \begin{bmatrix} \Sigma_{x_* x_*} & \Sigma_{X x_*}^T \\ \Sigma_{X x_*} & \Sigma \end{bmatrix}$. where $C' = \mathrm{diag}(y_1, \ldots, y_N)$.

The Proposed Method (Contd)



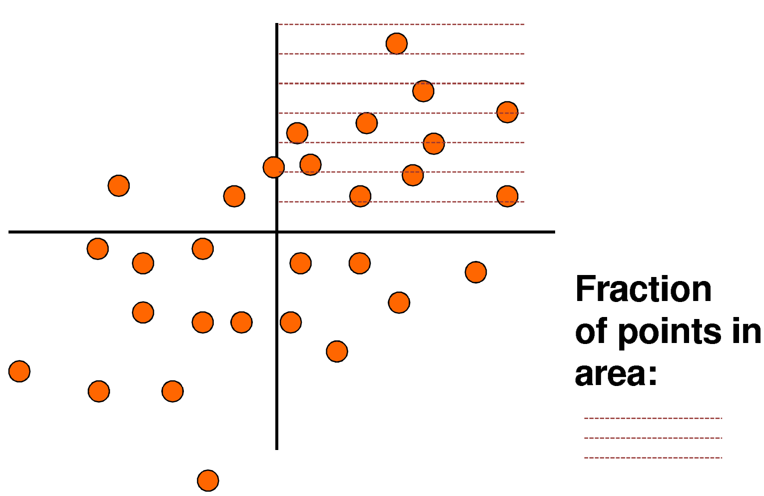
$$J_* = \frac{\int_{orth} \mathcal{N}\left(v, 0, I + A_{12}\Sigma' A_{12}\right) dv}{\int_{orth+} \mathcal{N}\left(v, 0, I + A_{12}\Sigma' A_{12}\right) dv} \equiv \frac{I_1}{I_2}$$

Multivariate Gaussian Integrals

- For high dimensionality it is a very hard problem.
- Generating points from the Gaussian distribution and counting the fraction that falls in integration area is not feasible.
- For example, consider an identity covariance matrix and a number N_{gen} of generated points.

$$\text{Fraction of points } \approx N_{gen} 2^{-N}$$

For $N = 100, \ N_{gen} = 100,000$, we get $7.9e-26$ points that fall in the integration area.



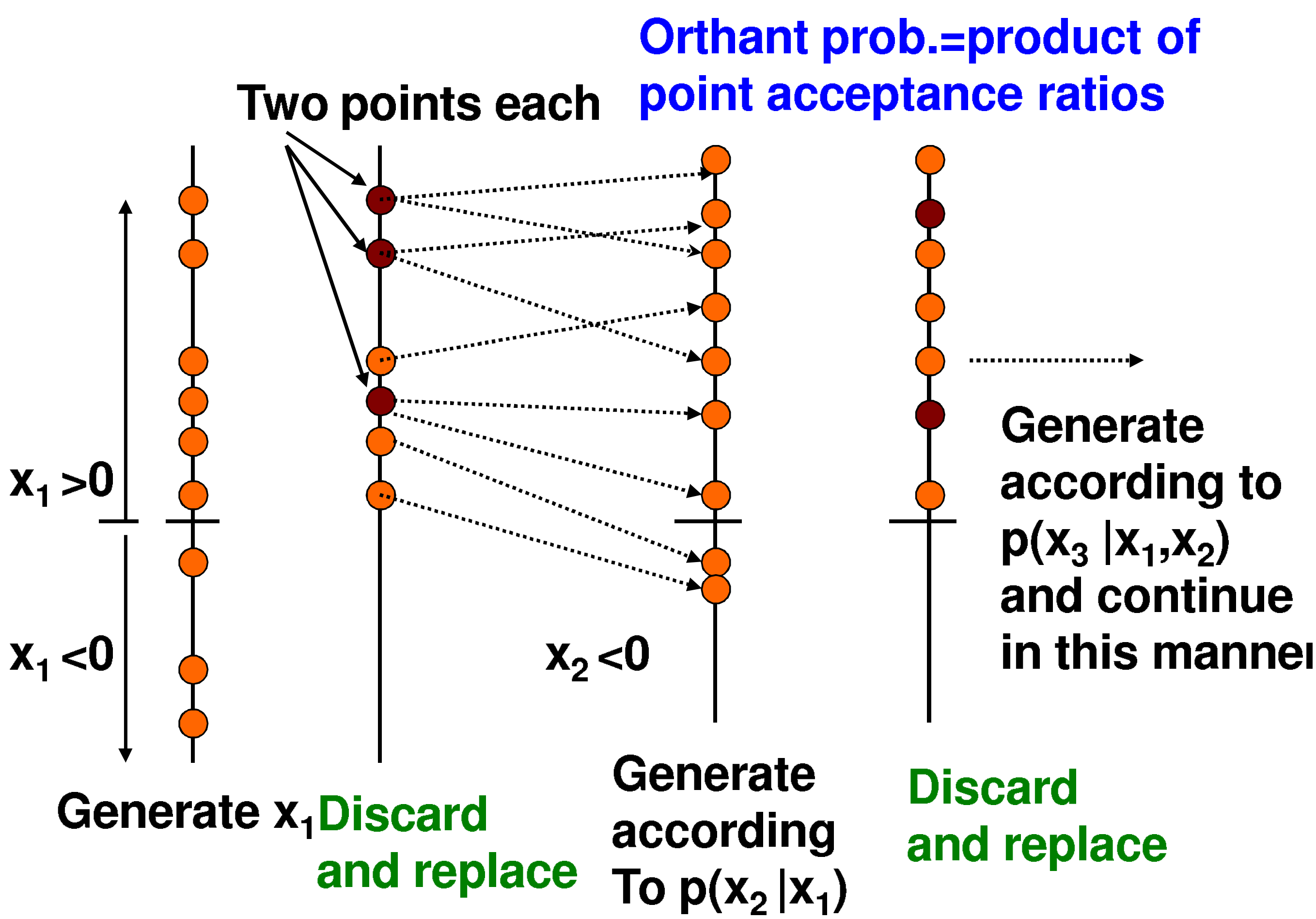
Proposed Integration Method

- The proposed new Monte Carlo method combines aspects of rejection sampling and bootstrap sampling.

- It can apply to any integration problem. As such, it is a new contribution for the general integration problem.

- Algorithm INTEG
  - We first generate samples for the first variable v_1.
  - Subsequently, we reject the points that fall outside the integral limits (for v_1).
  - We replenish in place of the discarded points by sampling with replacement from the existing points.
  - We move on to the second variable, v_2, and generate points using the conditional distribution $p(v_2|v_1)$ (conditioned on the v_1 points already generated).
  - Again, we reject the points of v_2 that fall outside the integration limit, and replenish by sampling with replacement.
  - We continue this manner until we reach the final variable v_N. The integral value is then estimated as the product of the acceptance ratios of the N variables.

Proposed Integration Method (Contd)



Properties of the Proposed Estimator

- We proved that it is a consistent estimator of the multivariate Gaussian integral (hence also of the posterior probability).

- This means that we can approach the true value by using enough generated points.

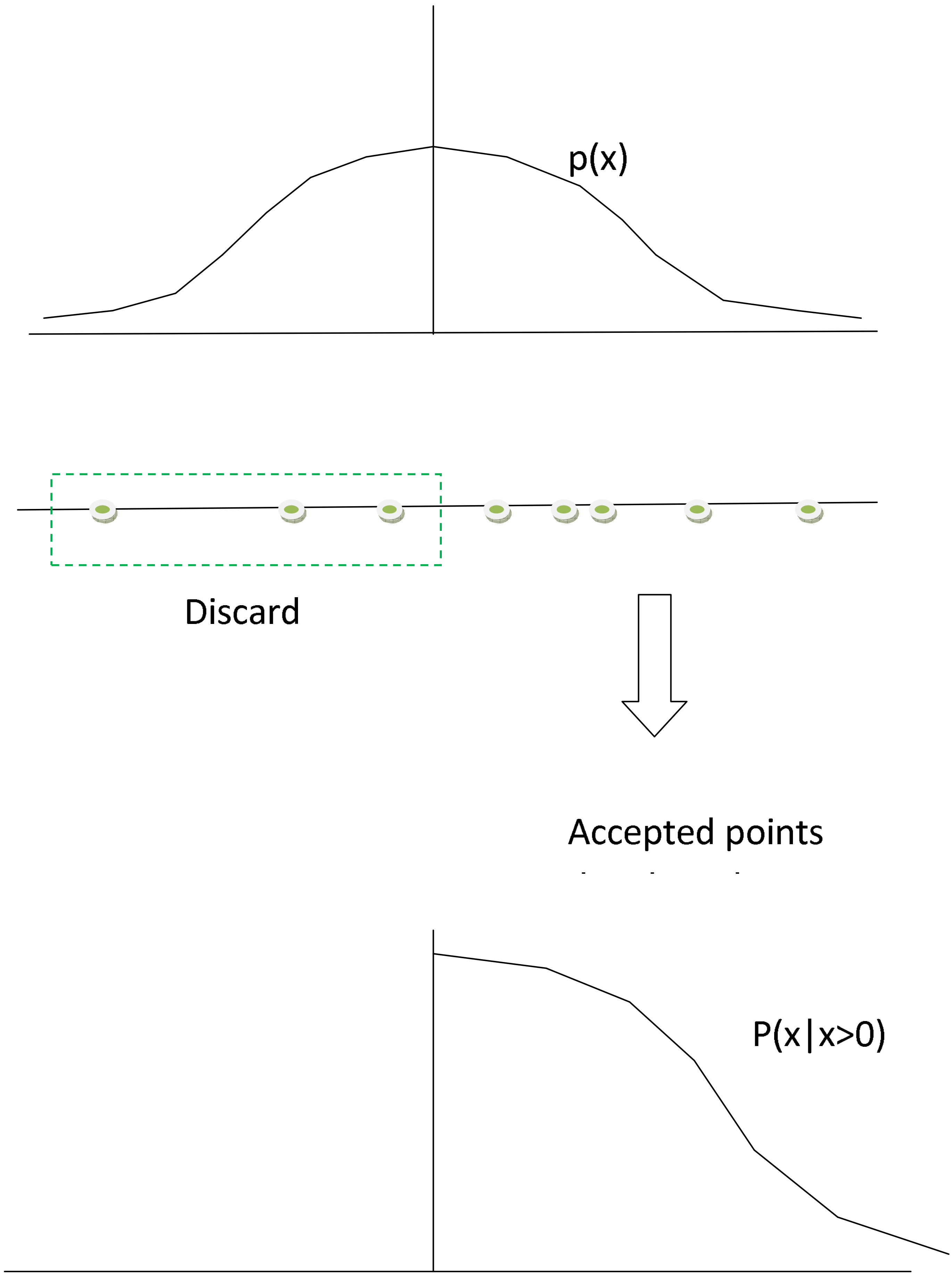
- The reason is as follows:
  - Assume the generated points v_i obey the distribution $p(v_i|v_{i-1} \geq 0, \ldots, v_1 \geq 0)$.
  - When we discard the points $v_i < 0$ and sample by replacement from the existing points, the points will be distributed as $p(v_i|v_i \geq 0, v_{i-1} \geq 0, \ldots, v_1 \geq 0)$.
  - When we generate the points v_{i+1} they will be distributed as $p(v_{i+1}|v_i \geq 0, \ldots, v_1 \geq 0)$.
  - Fraction accepted every step is about $P(v_i \geq 0|v_{i-1} \geq 0, \ldots, v_1 \geq 0)$.
  - Products of fractions accepted is about:

$$P(v_N \geq 0|v_{N-1} \geq 0, \ldots, v_1 \geq 0) \cdot$$
$$P(v_{N-1} \geq 0|v_{N-2} \geq 0, \ldots, v_1 \geq 0) \ldots$$
$$P(v_1 \geq 0)$$

which equals

$$P(v_N \geq 0, v_{N-1} \geq 0, \ldots, v_1 \geq 0)$$

AI Illustration of the Rejection Step



Mean Square Error of the Estimators (in Log Space)

- Let N be the dimension, N_G be the number of generated points, P_{orth} be the integral value, and $P_i \equiv P(x_i \geq 0 | x_{i-1} \geq 0, \ldots, x_1 \geq 0)$

- For the standard Monte Carlo:

$$\mathrm{MSE} = \frac{1 - P_{orth}}{P_{orth} N_G}$$

- For the new estimator:

$$\mathrm{MSE} = \frac{N}{N_G} \mathrm{Avg}\left(\frac{1 - P_i}{P_i} \right)$$

Numerical Example:

- Consider a 20-dimensional multivariate Gaussian distribution, with some specific covariance matrix.
- We applied both the new algorithm and the standard Monte Carlo method to evaluate the orthant integral $v \geq 0$.
- For both we used 100,000 generated values.
- For the standard Monte Carlo, no point fell in the area of integration.
- The true log integral equals -16.8587
- For the proposed algorithm, we obtained log(integral)= -16.8902 (0.19% error).

Other Approaches: Approximations to the Gaussian Integral

- In cases when we have a very large training set, e.g. in the thousands, we might opt for fast approximations for the sake of computation speed.

- We developed an approximation based on H. Joe (1995)'s Gaussian integral approximation.

- It is based on approximating the binary events $v_i \geq 0$ as Gaussian, and writing the joint Gaussian in terms of its conditional constituents.

$$
\begin{aligned}
J^* &= \frac{1}{2} + \frac{1}{2} \left(\tfrac{1}{4} - P_{N1} \quad \dots \quad \tfrac{1}{4} - P_{NN} \right) \cdot \\
& \begin{pmatrix} \frac{1}{4} & P_{12} - \frac{1}{4} & \dots & P_{1N} - \frac{1}{4} \\ P_{12} - \frac{1}{4} & \frac{1}{4} & \dots & P_{2N} - \frac{1}{4} \\ \vdots & \vdots & \vdots & \vdots \\ P_{1N} - \frac{1}{4} & P_{2N} - \frac{1}{4} & \dots & \frac{1}{4} \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}
\end{aligned}
$$

where P_{ij} is the bivariate centered Gaussian orthant integral for variables i and j. It can be analytically obtained using a simple formula.

Other Approximations: Linear Regression

- The multivariate Gaussian orthant integral is one of the very old problems that have defied any adequate solution (whether analytical or algorithmic.
- There exist a series expansion, but it is computationally intractable (exponential in N).
- Taking cue, we propose a series expansion. Instead of computing the coefficients analytically, we use a linear regression fit.
- We regress the orthant probability against the following possible homogeneous polynomials:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}, \quad \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}^2, \quad \sum_{i=1}^{N}\Big[\sum_{j=1}^{N} a_{ij}\Big]^2, \ldots$$

  where a_{ij} is the $(i,j)^{th}$ element of the inverse covariance matrix.

- How would we know the real orthant probabilities to obtain the regression coefficients:
- In the literature there are several special cases where a closed-form solution of the orthant probability exists. We use these to train the regression model.

Parameters that control smoothness

- In the prior distribution, the covariance for the latent variables is given by:

$$\text{cov}(f_i, f_j) = \beta e^{-\alpha \|x_i - x_j\|^2}$$

- α controls the degree of correlation among f_i and f_j.

- As such, it controls the the degree of smoothness of the f-surface.

- β controls the variance of the f_i's.

- It therefore controls how loose the connection is between the conditional mean of f_i and its resulting classification.

Marginal Likelihood

- A very potent way for the selection of these two parameters is to maximize the marginal likelihood function:

$$L = p(y|X) \equiv \int p(y|f)p(f|X)df$$

- It is a measure of how likely are the class memberships of the training data given the parameter values α and β.

- Find α and β that maximize L.

- We also proved that L is equivalent to a multivariate Gaussian orthant probability, that can be evaluated using the proposed methods.

Some Simulation Experiments

- We tested the new Monte Carlo algorithm on a special artificial classification problem, for which we can derive the "ground truth" probabilities.
- Convergence was achieved in every single run.
- There are no tuning parameters. In summary, the algorithm **works all the time**.
- In tens of tuning trials for the competing MCMC method, none converged.