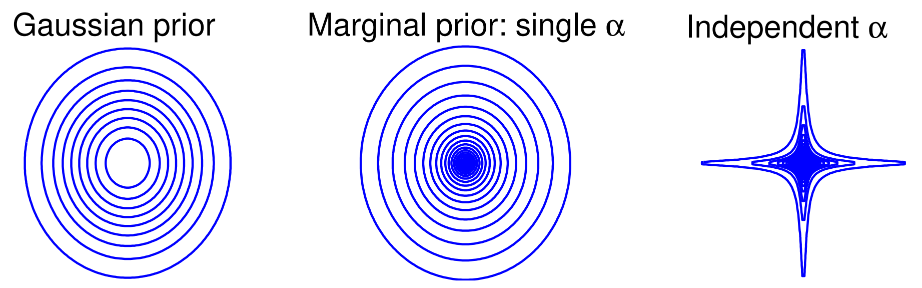
Bayesian Inference: Principles and Practice

3. Sparse Bayesian Models and the "Relevance Vector Machine"

Mike Tipping



Microsoft Research *Cambridge, UK*

Lecture 3: Overview*

- A hierarchical prior for ‘sparse’ linear models
- Sparse regression: (approximate) inference procedure
- Insight into sparsity
- Sparse classification: a further approximation
- Contrast with the support vector machine

*We won’t cover all the material in the notes in this lecture . . .

Bayes and Contemporary Machine Learning

- We've seen that marginalisation is a valuable component of the Bayesian data modelling paradigm

- We also saw that full Bayesian inference is often analytically intractable, although approximations for simple linear models could be very effective

- Historically, interest in Bayesian "machine learning" (but not statistics!) has focussed on approximations for *non-linear* models, *e.g.* neural networks:
  - The "evidence procedure" [MacKay]
  - Hybrid Monte Carlo sampling [Neal]

- Good news! — flexible *linear* kernel methods have become very fashionable, thanks mainly to the "support vector machine"

Linear Models and Sparsity

- Much interest in linear models has focused on *sparse* learning algorithms, which set many weights w_m to zero in the function $y(x) = \sum_m w_m \phi_m(x)$

- Sparsity offers:
  - Elegant complexity control
  - Feature extraction
  - Elucidation
  - Speed and compactness...

How To Encode Sparsity?

- In the algorithm:
  - Heuristic, ‘greedy’, sequential techniques (*e.g.* “matching pursuit”)

- In the objective function:

$$\widehat{E}(\mathbf{w}) = E_{\mathcal{D}}(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

  - usually via the penalty term $E_W(\mathbf{w})$
  - but can be via the data error term $E_{\mathcal{D}}(\mathbf{w})$ (*e.g.* the SVM)

- In the prior:

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})\ p(\mathbf{w})}{p(\mathcal{D})}$$

  - Recall: correspondence between $E_W(\mathbf{w})$ and $-\log p(\mathbf{w})$

Penalty Terms & Priors

- Most common regularisation term: $E_W(\mathbf{w}) = \sum_{m=1}^{M} |w_m|^2$
  - Corresponds to a Gaussian prior $p(\mathbf{w}) \propto \exp\left(-\sum_m |w_m|^2\right)$
  - Easy to work with & an effective regulariser, but no sparsity pressure

- 'Correct' term would be $E_W(\mathbf{w}) = \sum_m |w_m|^0$, but this is very ugly to work with!

- Instead, $E_W(\mathbf{w}) = \sum_m |w_m|^1$ is a workable compromise which gives reasonable sparsity and reasonable tractability:
  - "LP-machines"
  - "Basis pursuit"
  - "Bayesian Pruning", as a Laplace prior $p(\mathbf{w}) \propto \exp\left(-\sum_m |w_m|\right)$

A Sparse Bayesian Prior

- In fact, we *can* obtain sparsity by retaining the traditional Gaussian prior
  - This is great news for tractability
  - Furthermore, this approach leads to *scale invariance*

- The modification to our earlier Gaussian prior is subtle:

$$p(\mathbf{w}|\alpha_1, \ldots, \alpha_M) = (2\pi)^{-M/2} \prod_{m=1}^{M} \alpha_m{}^{1/2} \exp\left\{-\frac{1}{2}\sum_{m=1}^{M} \alpha_m w_m^2\right\}$$

- What's new? We now have M hyperparameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_M)$, one α_m independently controlling the (inverse) variance of each weight w_m

Hierarchical Priors

- The prior $p(\mathbf{w}|\boldsymbol{\alpha})$ is Gaussian, but conditioned on $\boldsymbol{\alpha}$
- We should now define hyperpriors over all α_m
- Previously, we utilised a log-uniform hyperprior — this is a special case of a more general *Gamma* hyperprior
- This gives a *hierarchical* prior: we must marginalise to see its 'true' form

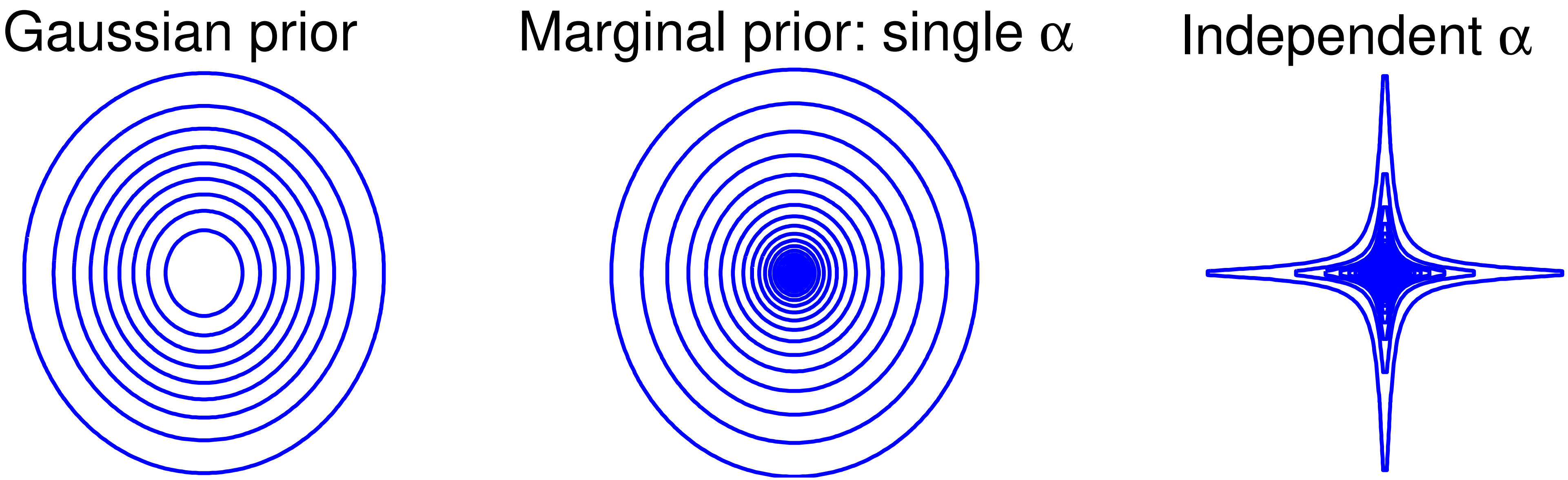
$$p(w_m) = \int p(w_m|\alpha_m)\, p(\alpha_m)\, d\alpha_m$$

- For a Gamma $p(\alpha_m)$, $p(w_m)$ is a *Student-t* distribution
- Its equivalent as a penalty function would be: $\sum_m \log |w_m|$

Priors Compared

- Specifying independent hyperparameters α_m is the key to sparsity

- Example marginal priors $p(w_1, w_2)$ illustrated below:



A Sparse Bayesian Model for Regression

- As before: independent Gaussian noise: $t_n \sim N(y(\mathbf{x}_n; \mathbf{w}), \sigma^2)$

- This gives a corresponding likelihood:

$$p(\mathbf{t}|\mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-N/2} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{t} - \mathbf{\Phi}\mathbf{w}\|^2\right\}$$

  - $\mathbf{t} = (t_1 \ldots t_N)^{\mathrm{T}}$
  - $\mathbf{w} = (w_1 \ldots w_M)^{\mathrm{T}}$
  - $\mathbf{\Phi}$ is the $N \times M$ 'design' matrix with $\mathbf{\Phi}_{nm} = \phi_m(\mathbf{x}_n)$

Approximate Inference

- Desire posterior over all unknowns:

$$p(\mathbf{w}, \alpha, \sigma^2|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w}, \alpha, \sigma^2)p(\mathbf{w}, \alpha, \sigma^2)}{p(\mathbf{t})}$$

- Can't compute analytically! So as previously, decompose as:

$$p(\mathbf{w}, \alpha, \sigma^2|\mathbf{t}) \equiv p(\mathbf{w}|\mathbf{t}, \alpha, \sigma^2)\ p(\alpha, \sigma^2|\mathbf{t})$$

  - $p(\mathbf{w}|\mathbf{t}, \alpha, \sigma^2)$, the 'weight posterior' distribution, is tractable
  - $p(\alpha, \sigma^2|\mathbf{t})$ must be approximated

The Weight Posterior Term

- Given the data, the posterior distribution over weights is Gaussian:

$$p(\mathbf{w}|\mathbf{t},\boldsymbol{\alpha},\sigma^2) = \frac{p(\mathbf{t}|\mathbf{w},\sigma^2)\,p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha},\sigma^2)}$$

$$= (2\pi)^{-(N+1)/2}|\boldsymbol{\Sigma}|^{-1/2}\exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\}$$

- with
  - $\boldsymbol{\Sigma} = (\sigma^{-2}\boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{\Phi} + \mathbf{A})^{-1}$
  - $\boldsymbol{\mu} = \sigma^{-2}\boldsymbol{\Sigma}\boldsymbol{\Phi}^{\mathsf{T}}\mathbf{t}$
  - $\mathbf{A} = \mathrm{diag}(\alpha_1, \alpha_2, \ldots, \alpha_M)$

- **Key point:** if $\alpha_m = \infty$, the corresponding $\mu_m = 0$

The Hyperparameter Posterior Term

- As before, for uniform hyperpriors over $\log \alpha$ and $\log \sigma$, $p(\boldsymbol{\alpha}, \sigma^2 | \mathbf{t}) \propto p(\mathbf{t} | \boldsymbol{\alpha}, \sigma^2)$

- Integrate out weights to obtain the *marginal likelihood*:

$$
\begin{aligned}
p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) &= \int p(\mathbf{t}|\mathbf{w}, \sigma^2)\, p(\mathbf{w}|\boldsymbol{\alpha})\, d\mathbf{w}, \\
&= (2\pi)^{-N/2} |\sigma^2 \mathbf{I} + \boldsymbol{\Phi} \mathbf{A}^{-1} \boldsymbol{\Phi}^{\mathsf{T}}|^{-1/2} \exp\left\{ -\frac{1}{2} \mathbf{t}^{\mathsf{T}} (\sigma^2 \mathbf{I} + \boldsymbol{\Phi} \mathbf{A}^{-1} \boldsymbol{\Phi}^{\mathsf{T}})^{-1} \mathbf{t} \right\}
\end{aligned}
$$

- We maximise $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)$ to find $\boldsymbol{\alpha}_{\mathrm{MP}}$ and σ^2_{MP} ("type-II maximum likelihood")

- In Lecture 2, we found the single α_{MP} empirically

- For multiple hyperparameters, we must optimise $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)$ directly

Hyperparameter Re-estimation

- Differentiating $\log p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)$ gives iterative re-estimation formulae:

$$\alpha_i^{\text{new}} = \frac{\gamma_i}{\mu_i^2}$$

$$(\sigma^2)^{\text{new}} = \frac{\|\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{\mu}\|^2}{N - \sum_{i=1}^{M} \gamma_i}$$

- For convenience we have defined the quantities

$$\gamma_i = 1 - \alpha_i \boldsymbol{\Sigma}_{ii}$$

- $\gamma_i \in [0, 1]$ is a measure of 'well-determinedness' of parameter w_i

Summary of Inference Procedure

❶ Initialise $\{\alpha_i\}$ and σ^2 (or fix latter if known)

❷ Compute weight posterior sufficient statistics $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

❸ Compute all $\{\gamma_i\}$, then re-estimate $\{\alpha_i\}$ (and σ^2 if desired)

❹ Repeat from ❷ until convergence

❺ Make predictions for new data via the predictive distribution:

$$p(t_*|\mathbf{t}) = \int p(t_*|\mathbf{w}, \sigma^2_{\mathrm{MP}})\, p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}})\, d\mathbf{w}$$

the mean of which is $y(\mathbf{x}_*; \boldsymbol{\mu})$

❻ Can 'delete' basis functions for which optimal $\alpha_i = \infty$, since $\mu_i = 0$

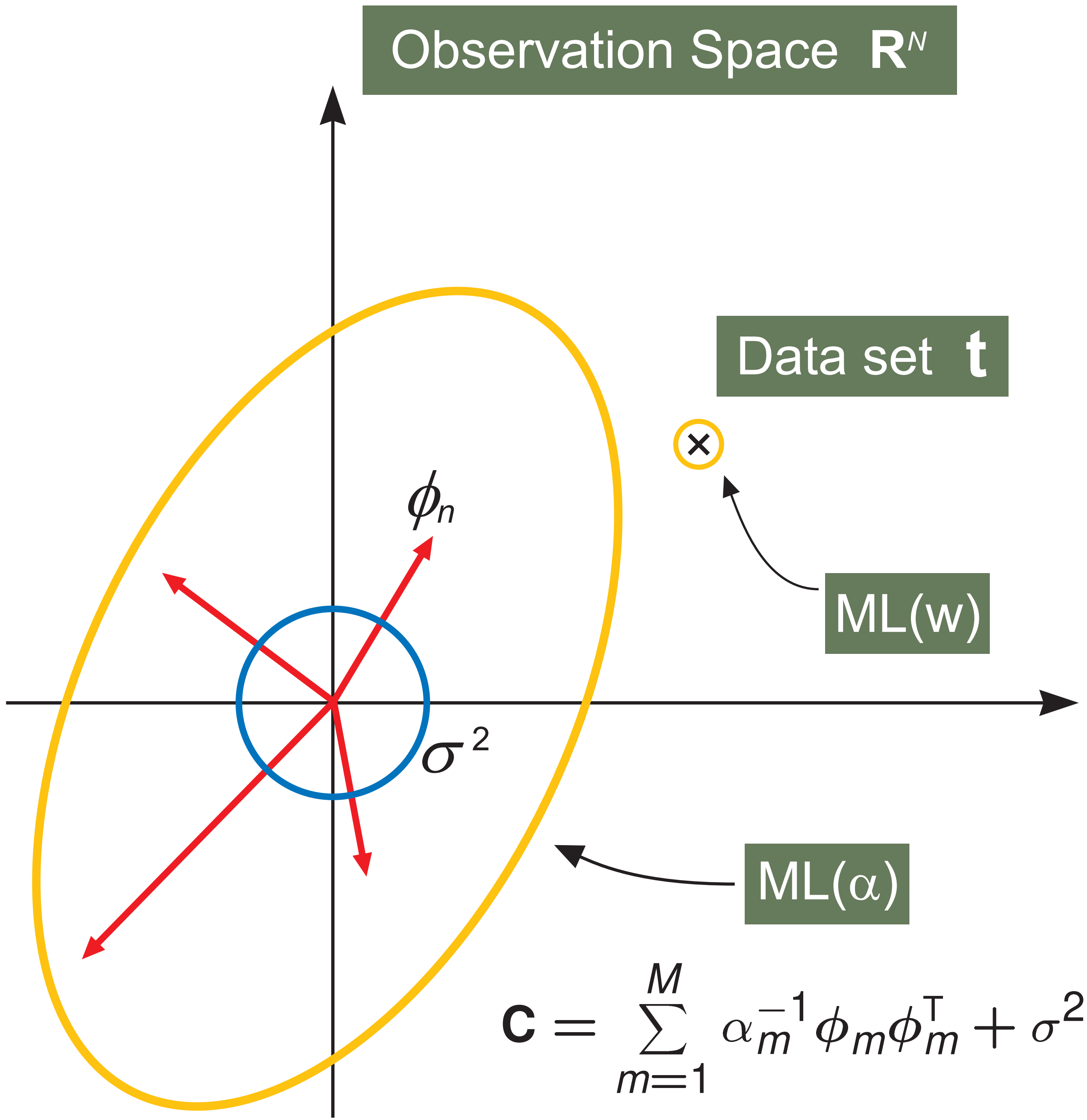
Sparsity: Algorithmic Insight

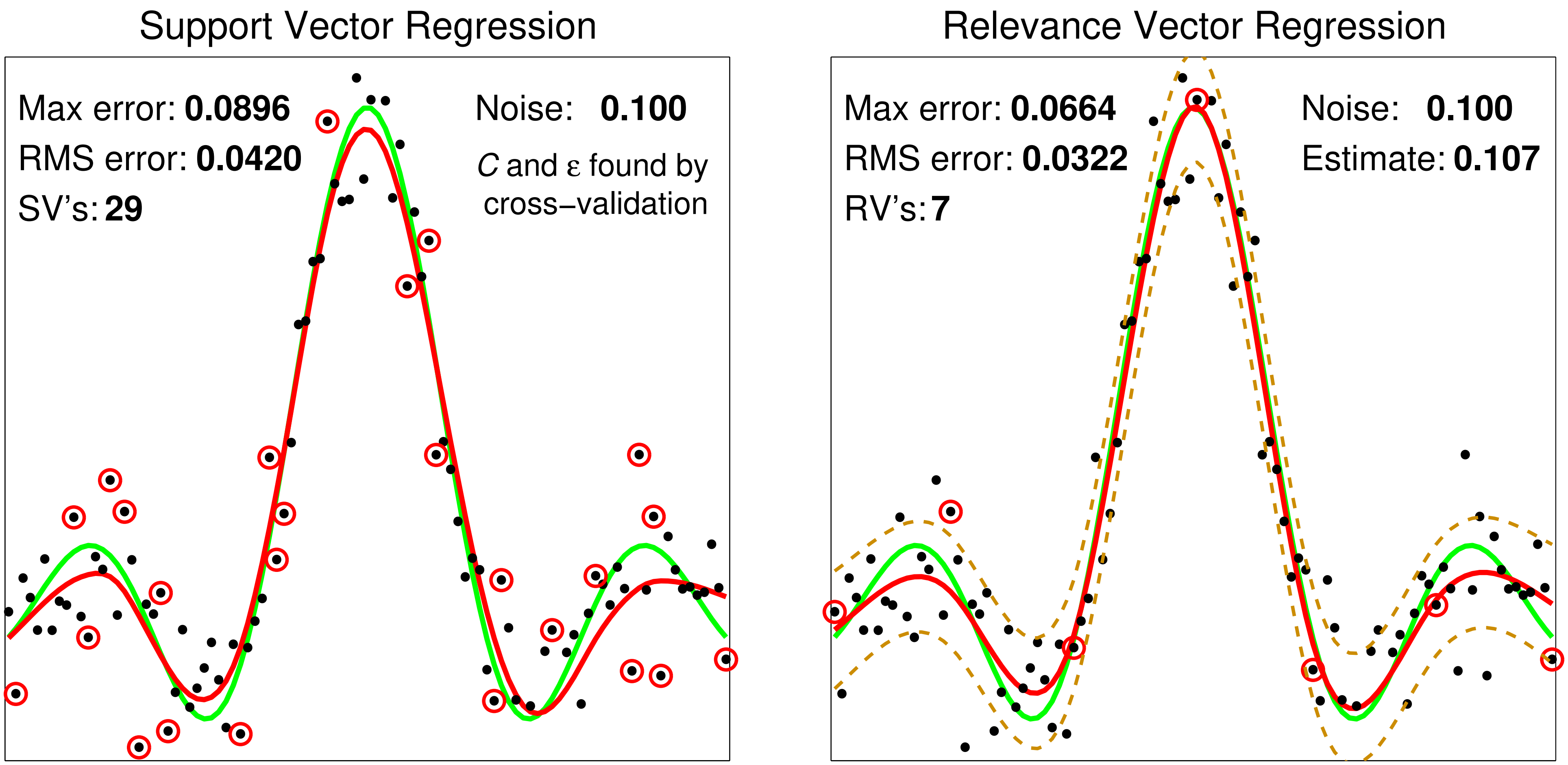
- ML($\mathbf{w}$): conventional maximum likelihood (or least-squares)
  - optimises $M+1$ parameters, $\mathbf{w}$ and σ^2
  - Since $\sigma^2 \to 0$, $p(\mathbf{t}|\mathbf{w})$ is a δ-function located at $\mathbf{t}$
  - No “Ockham penalty” !

- ML($\boldsymbol{\alpha}$): maximising the marginal likelihood
  - optimises $M+1$ parameters, $\boldsymbol{\alpha}$ and σ^2
  - $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)$ is a zero-mean *Gaussian process* with covariance

$$\mathbf{C} = \sum_{m=1}^{M} \alpha_m^{-1} \boldsymbol{\phi}_m \boldsymbol{\phi}_m^{\mathrm{T}} + \sigma^2 \mathbf{I}$$

where $\boldsymbol{\phi}_m = [\phi_m(\mathbf{x}_1), \ldots, \phi_m(\mathbf{x}_N)]^{\mathrm{T}}$


Observation Space $\mathbf{R}^N$
Data set $\mathbf{t}$
ϕ_n
σ^2
ML(w)
ML(α)
$\mathbf{C} = \sum_{m=1}^{M} \alpha_m^{-1} \phi_m \phi_m^{\mathrm{T}} + \sigma^2 \mathbf{I}$



Support Vector Regression
Max error: **0.0896**
RMS error: **0.0420**
SV's: **29**
Noise: **0.100**
C and ε found by cross–validation
Relevance Vector Regression
Max error: **0.0664**
RMS error: **0.0322**
RV's: **7**
Noise: **0.100**
Estimate: **0.107**

Sparse Bayesian Classification

- The likelihood is now *Bernoulli*, rather than Gaussian:

$$P(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} g\{y(\mathbf{x}_n;\mathbf{w})\}^{t_n} \left[1 - g\{y(\mathbf{x}_n;\mathbf{w})\}\right]^{1-t_n}$$

  with $g(y) = 1/(1+e^{-y})$ the 'logistic sigmoid' function

- Note: no noise variance σ^2

- Same sparse prior $p(\mathbf{w}|\boldsymbol{\alpha})$ as for regression

- Unlike for regression and the Gaussian likelihood model, $p(\mathbf{w}|\mathbf{t},\boldsymbol{\alpha})$ cannot be obtained analytically, we so utilise a Laplace (Gaussian) approximation

Classification: Gaussian Posterior Approximation

- For the current values of $\boldsymbol{\alpha}$, find the posterior mode $\mathbf{w}_{\mathrm{MP}}$ via optimisation

- Compute the Hessian at $\mathbf{w}_{\mathrm{MP}}$:

$$\nabla_{\mathbf{w}} \nabla_{\mathbf{w}} \log p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha})\big|_{\mathbf{w}_{\mathrm{MP}}} = -(\boldsymbol{\Phi}^{\mathrm{T}} \mathbf{B} \boldsymbol{\Phi} + \mathbf{A})$$

  where $\mathbf{B}_{nn} = g\{y(\mathbf{x}_n; \mathbf{w}_{\mathrm{MP}})\} \left[1 - g\{y(\mathbf{x}_n; \mathbf{w}_{\mathrm{MP}})\}\right]$

- Negate and invert to give the covariance $\boldsymbol{\Sigma}$ for a Gaussian approximation $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}) \approx N(\mathbf{w}_{\mathrm{MP}}, \boldsymbol{\Sigma})$

- Hyperparameters $\boldsymbol{\alpha}$ are updated as before using the approximated $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

- Note that $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha})$ is log-concave

Support Vector Machines

- A state-of-the-art method for classification and regression
- Given data set comprising N input vectors $\mathbf{x}_n$, model has form

$$y(\mathbf{x};\mathbf{w}) = \sum_{n=1}^{N} w_n K(\mathbf{x},\mathbf{x}_n) + w_0$$

- As many *kernel* functions $K(\cdot,\mathbf{x}_n)$ as examples $\Rightarrow M = N + 1$ parameters
- Support vector learning: minimise objective function of form:

$$\widehat{E}(\mathbf{w}) = E_{\mathcal{D}}(\mathbf{w}) - \lambda \times (\text{size of margin})$$

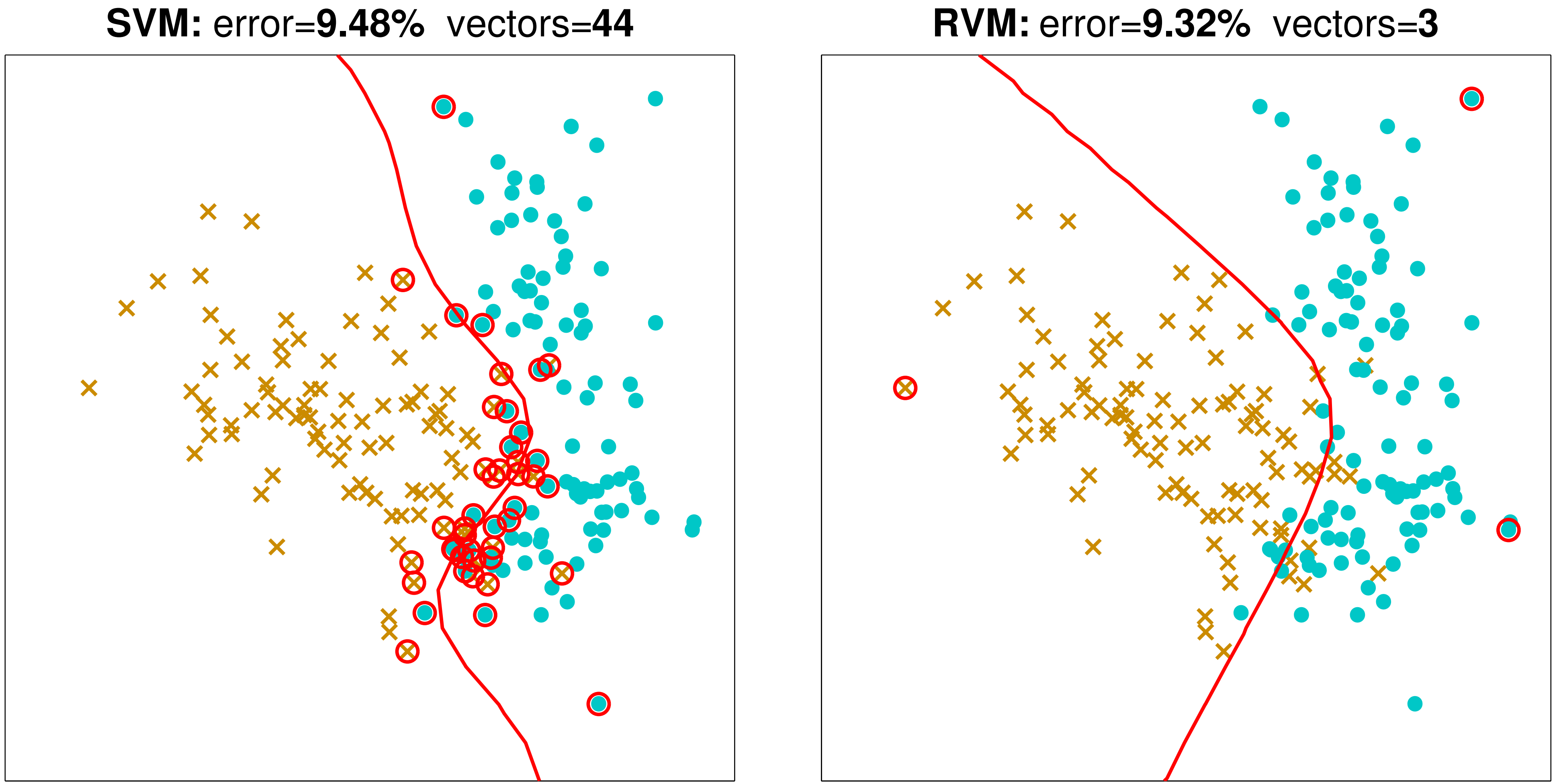
  - gives excellent accuracy (particularly in classification)
  - as a side-effect, many w_n get set to zero — the model is *sparse*

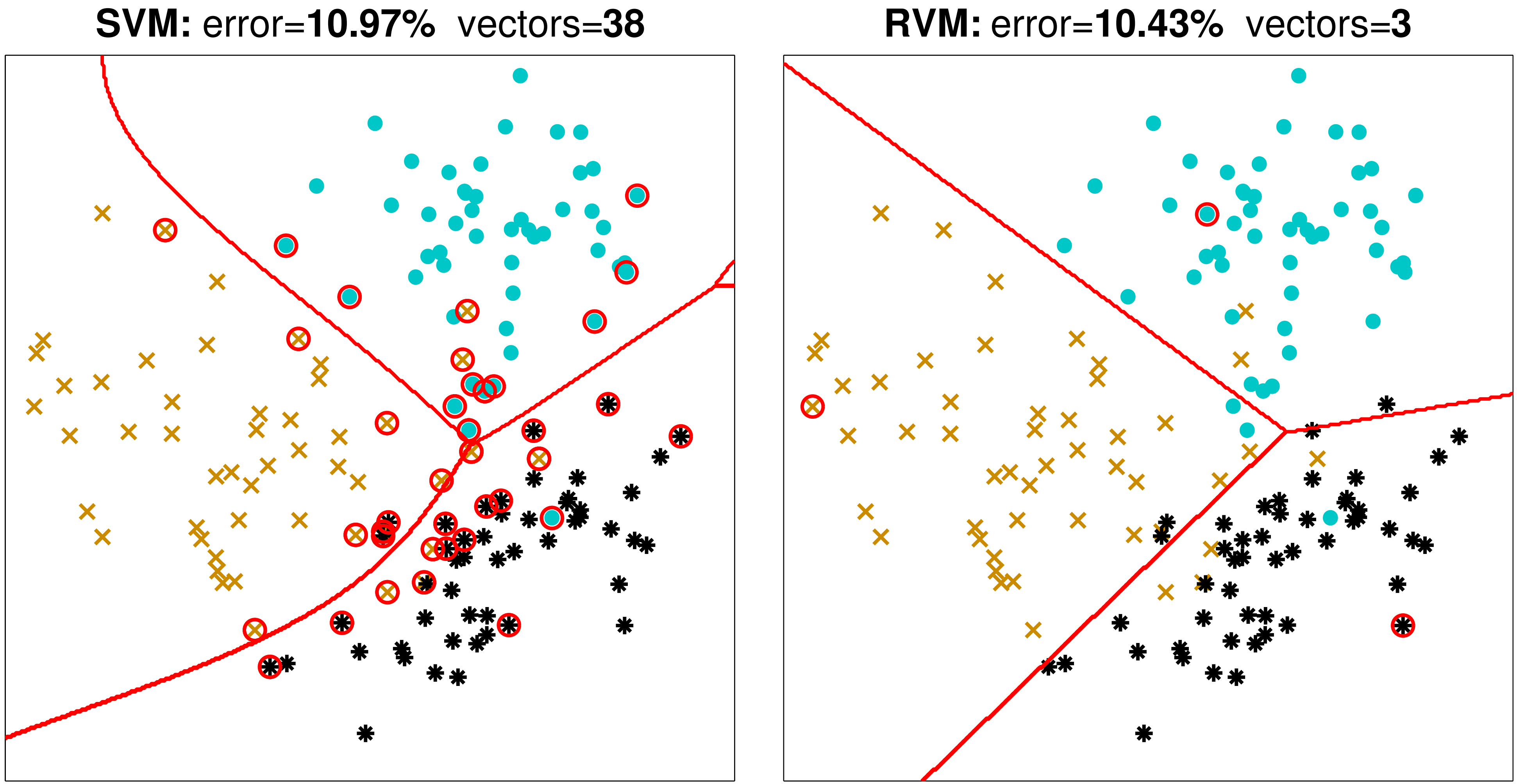
The “Relevance Vector Machine” (RVM)

- The RVM is simply a sparse Bayesian model utilising the same data-dependent kernel basis as the SVM:

$$y(\mathbf{x}; \mathbf{w}) = \sum_{n=1}^{N} w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$$

- Demonstration of RVM classification ...


SVM: error=**9.48%** vectors=**44**
RVM: error=**9.32%** vectors=**3**


SVM: error=10.97% vectors=38
RVM: error=10.43% vectors=3

Comparison with the SVM

- General observations:
  - RVM gives better generalisation in regression (?)
  - SVM gives better generalisation in classification (?)
  - RVM is much sparser (but the SVM is not designed to be sparse)

- There are other advantages of a Bayesian approach:
  - no ‘nuisance’ parameters to set
  - posterior probabilities in classification
  - error bars in regression
  - principled method for more than two classes
  - not limited to Mercer kernels
  - potential to estimate input scale parameters and compare kernels

Choosing the Kernel

- As in the SVM, we must choose the kernel and set any associated parameter(s)
- A Bayesian could compare alternative kernels by computing the fully marginalised probabilities of the data under candidate models. *e.g.*:

$$p(\mathbf{t}|\mathcal{K}_1) = \int p(\mathbf{t}|\boldsymbol{\alpha}, \mathcal{K}_1)\, p(\boldsymbol{\alpha}|\mathcal{K}_1)\, d\boldsymbol{\alpha}$$

- We already know this integral isn't analytically tractable
- Approximation via sampling (as in Lecture 2) is not feasible for multiple $\boldsymbol{\alpha}$
- Deterministic approximations to this integral have proved inaccurate
- But $p(\mathbf{t}|\boldsymbol{\alpha}_{\mathrm{MP}}, \mathcal{K})$ is a 'reasonable' criterion for choosing kernels

Kernel ‘Input Scale’ Parameters

- For example, consider choice of η in $K(\mathbf{x}, \mathbf{x}_n) = \exp\left\{-\eta\|\mathbf{x} - \mathbf{x}_n\|^2\right\}$

- SVM: cross-validation can be used to set scale parameter η

- RVM: we can *optimise* the marginal likelihood function with respect to η

- Furthermore, we can optimise multiple scale parameters $\boldsymbol{\eta}$, one for each of the *d* input dimensions:

$$K(\mathbf{x}, \mathbf{x}_n) = \exp\left\{-\sum_{k=1}^{d} \eta_k (x_k - x_{nk})^2\right\}$$

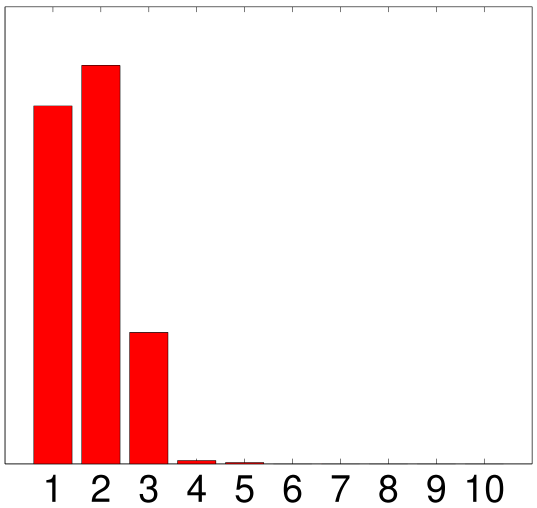
- Implementing sparsity of *input* variables (*q.v.* Gaussian process models)

Impact on Regression Benchmarks

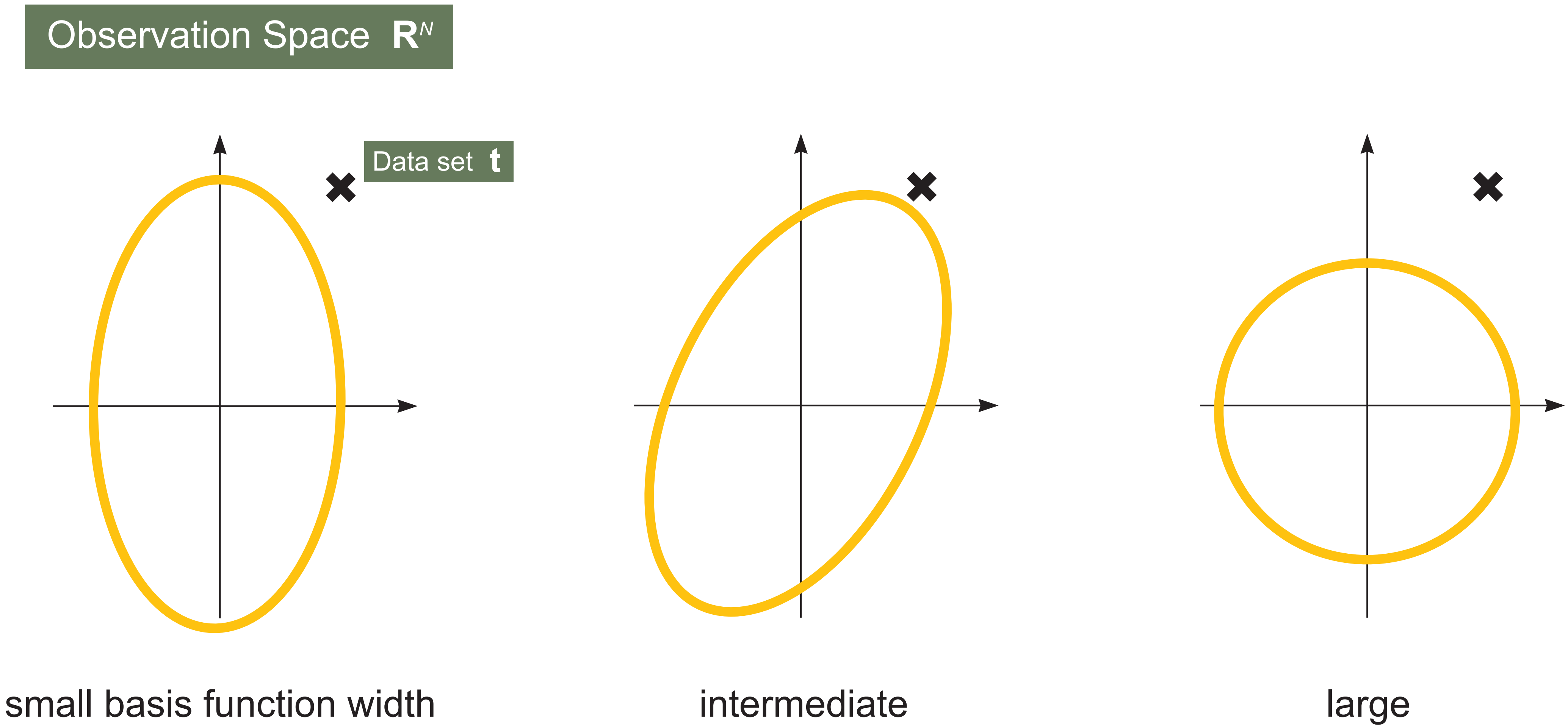
- η-RVR: optimisation over both $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$

	Test error			# kernels		
Dataset	**SVR**	**RVR**	**η-RVR**	**SVR**	**RVR**	**η-RVR**
Friedman #1	2.92	2.80	0.27	116.6	59.4	11.5
Friedman #2	4140	3505	2593	110.3	6.9	3.9
Friedman #3	0.0202	0.0164	0.0119	106.5	11.5	6.4

- Friedman #1: 10-dimensional input space, but function depends only on variables 1–5. Final η-values shown on right:



Ockham’s Razor and Kernel Width



Summary

- Experience appears to show that the sparse Bayesian learning procedure works highly effectively:
  - generalisation performance is typically very good
  - models are typically extremely (near-optimally?) sparse

- Challenge: obtain a reliable approximation to integrating out $\boldsymbol{\alpha}$ (for model comparison)

- In Lecture 4, we'll look at:
  - Properties of the marginal likelihood function
  - An efficient algorithm for optimising $\boldsymbol{\alpha}$
  - Extensions and applications of sparse Bayesian models

Regression Performance Illustration

Data set	*N*	*d*	*errors* SVM	*errors* RVM	*vectors* SVM	*vectors* RVM
Sinc (Gaussian noise)	100	1	0.378	0.326	45.2	6.7
Sinc (Uniform noise)	100	1	0.215	0.187	44.3	7.0
Friedman #1	240	10	2.92	2.80	116.6	59.4
Friedman #2	240	4	4140	3505	110.3	6.9
Friedman #3	240	4	0.0202	0.0164	106.5	11.5
Boston Housing	481	13	8.04	7.46	142.8	39.0
Normalised Mean			1.00	**0.86**	1.00	**0.15**

Classification Performance Illustration

			errors		vectors	
Data set	***N***	***d***	**SVM**	**RVM**	**SVM**	**RVM**
Pima Diabetes	200	8	20.1%	19.6%	109	4
U.S.P.S.	7291	256	4.4%	5.1%	2540	316
Banana	400	2	10.9%	10.8%	135.2	11.4
Breast Cancer	200	9	26.9%	29.9%	116.7	6.3
Titanic	150	3	22.1%	23.0%	93.7	65.3
Waveform	400	21	10.3%	10.9%	146.4	14.6
German	700	20	22.6%	22.2%	411.2	12.5
Image	1300	18	3.0%	3.9 %	166.6	34.6
Normalised Mean			1.00	**1.08**	1.00	**0.17**