Deconvolution and Classification

Raymond J. Carroll
Department of Statistics
Member, Center for Statistical Bioinformatics
Director, Institute for Applied Mathematics and Computational Science
Texas A&M University
http://stat.tamu.edu/~carroll
Other Acknowledgments

• I gratefully acknowledge financial support from the U. S. National Cancer Institute (R37-CA057030), the National Science Foundation-DTRA (Mallick, P.I.) and the King Abdullah University of Science and Technology (KAUST, Award Number KUS-CI-016-04).

• Many thanks to the organizers and to Lola Ugarte and Ana Militino for inviting me to Pamplona.
College Station, home of Texas A&M University

Palo Duro Canyon, the Grand Canyon of Texas

Big Bend National Park

Guadalupe Mountains National Park

Wichita Falls, that's my hometown

East Texas 😞

Wichita Falls, Wichita Falls, that's my hometown

West Texas 😊

College Station, home of Texas A&M University
Palo Duro Canyon of the Red River
Main Conclusion About Classification

• I am interested in classification

• Engineers want to “deconvolve, reduce dimension and then classify”

• I claim that rather than blindly always do deconvolution, one should also consider a strategy of “skip the deconvolution step”
Frequency Agile Lidar Data

• This is a recent project at Texas A&M from DTRA grants

• Here is a comic describing the process
Frequency Agile Lidar Data

- The signal is **transmitted** in a series of **bursts**, and for each burst, a signal is **received** across **time** and for various **wavelengths**
LIDAR Data

Bani Mallick  Swarup De  Xiaolei Xun

Peter Hall  Aurore Delaigle
Frequency Agile Lidar Data

• There is a transmitted signal
• There is background
• There is a received signal, which is then background corrected
• For each burst (1000, first 200 are background), wavelength (19) and sample (28), we see 625 observations across time, i.e., equally spaced functional data with noise.
• “Pretty Large Data”: 11,875,000 per sample
There are two types of signals:

1. The first is benign, ordinary dust-like stuff that has been released.
2. The second are biological aerosols.
3. The DTRA wants a method to classify which type is out there.
The data for one sample, one wavelength, as a function of burst and time (range)

There are many more bursts and many more time periods
Frequency Agile Lidar Data

The data for one sample, one wavelength, as a function of a few bursts and selected time points (range)
Frequency Agile Lidar Data

The data for one sample, one wavelength, as a function of a few time points (range) and selected bursts.
Frequency Agile Lidar Data

Four samples and same burst and wavelength.

Background corrected, truncated at zero and normalized by peak height.

There are many ways to normalize
Each sample normalized to have maximum = 1.0 in the central range across all bursts and wavelengths

Then averaged over bursts and wavelengths
Some hint that some classification might be possible.

A 20% error rate if you reduce everything to simply the maximum of the sample.
Error Rate Estimation

• For this talk, I use leave-one-out error rate estimation

• This means I take out each sample, then form the predictor from the rest. I then predict the left out sample, and see if I am correct.

• Then I get the average error rate across the samples

• There are many other methods to do this too
Data

- aerosol type $a = 1, 2$
- sample $i = 1, \ldots, 14$ within type
- $b =$ burst,
- $w =$ wavelength
- Within each burst and wavelength, we see a function across $t = \text{time}$

$$R_{\text{aiwb}} (t)$$
Data

• Use 800 non-background bursts, 19 wavelength and, using the center of the time points, 250 time points.

• This is 3,800,000 observations per sample

• There are only 28 samples

• The scale (say maximum) of the data varies depending on bursts and wavelengths
Data

• For aerosol type \( a = 1,2 \), and sample \( i=1,\ldots,n \) within type, we observe \( R_{aiwb}(t) \).

• The investigators want to classify.

• **Method 1**: Simply use the observed data and fit your **favorite classifier** (LDA, SVM, Centroids, ...) and use your **favorite normalization** to get things on the same scale.
Approaches

• What sets this problem apart is that there is a convolution/deconvolution aspect to this problem.

• Along with the received signal $R_{aiwb}(t)$, there is the transmitted signal $T_{aiwb}(t)$.

• There is thought to be a true signal $G_{aiwb}(t)$. 
Approaches

• The convolution equation is

\[ R_{aiwb}(t) = \int_0^x G_{aiwb}(z)T_{aiwb}(t-z) \, dz + \varepsilon_{aiwb}(t) \]

• Here, \( \varepsilon_{aiwb}(t) \) is supposed to be a Gaussian process

• Should one use \( R_{aiwb}(t) \) or deconvolve and use \( \hat{G}_{aiwb}(y) \)?
The deconvolution equation is

\[ R_{aiwb}(t) = \int_0^x G_{aiwb}(z) T_{aiwb}(t-z) \, dz + \varepsilon_{aiwb}(t) \]

Obtaining \( \hat{G}_{aiwb}(t) \) is based on a Wiener-Helstrom deconvolution method.
Lidar Data Algorithm

\[ R_{aiwb}(t) = \int_{0}^{x} G_{aiwb}(z) T_{aiwb}(t-z) \, dz + \varepsilon_{aiwb}(t) \]

- You first have to estimate the spectral density of \( \varepsilon_{aiwb}(t) \).
- We have 200 bursts where there is background only and \( G = 0 \), so this is just spectral density estimation using multiple time series.
Lidar Data Algorithm

\[
R_{\text{aiwb}}(t) = \int_0^x G_{\text{aiwb}}(z) T_{\text{aiwb}}(t-z) \, dz + \varepsilon_{\text{aiwb}}(t)
\]

• You next estimate the spectral density of \( G_{\text{aiwb}}(t) \).

• There is no real data to do this. The literature assumes that \( G_{\text{aiwb}}(t) \) is Gaussian and that the spectral density depends on 2 parameters.

• The two regularization parameters are not estimated: they are picked to give “visually pleasing results”.
Lidar Data Algorithm

• There are data-dependent ways to pick the two regularization parameters

• One is to minimize some version of an error rate
Frequency Agile Lidar Data

Each sample normalized to have maximum = 1.0 in the central range across all bursts and wavelengths.

Then averaged over bursts and wavelengths.
Deconvolved: does not look much better

Actually, looks worse.

Plus, much more noise
The Lidar Data

• We have done various things to implement classification on the basis of \( R_{aiwb}(t) \) versus \( \hat{G}_{aiwb}(t) \)

• Different classifiers, different schemes for normalizing peak height, etc.

• In all cases, the leave-one-out error rate was higher for deconvolution
The Lidar Data

• The error rates were roughly at best 18% for using the observed data when I used standard classifiers.

• There were roughly at best 30% for using the deconvolved data.
The Lidar Data

• However, the data sets are quite large, and it would seem to make some sense to do some sort of dimension reduction.

• Engineering considerations suggest that

\[ G_{\text{aibw}}(t) = \rho_{\text{aibw}} C_{\text{aib}}(t) \]

• Here, \( \rho_{\text{aibw}} \) is the “spectral backscatter”, which will be estimated, averaged over the bursts, and then used for classification. Does not depend on time!
The Lidar Data

• Here, $\rho_{aibw}$ is the "spectral backscatter"

• It is normalized to be non-negative and to have norm $= 1$ over the bursts

• It is estimated by alternating regressions

• The special form $G_{aibw}(t) = \rho_{aibw} C_{aib}(t)$ allows alternating regressions, done separately for each sample
The Lidar Data

• Here, \( G_{aibw}(t) = \rho_{aibw} C_{aib}(t) \)

• For each burst and time, regress \( G \) on the spectral backscatter to estimate \( C \), and make \( C \) nonnegative.

• For each wave and burst, regress \( G \) on \( C \) to estimate the spectral backscatter. Make nonnegative and then make norm over bursts = 1

• Iterate
The Lidar Data

- The resulting spectral backscatter estimates **derived from deconvolution** are then fed into a Linear SVM for classification.

- What do we do with the **observed data**?
The Lidar Data

• Engineering considerations suggest that

\[ G_{aibw}(t) = \rho_{aibw} C_{aib}(t) \]

• But the **observed data have the same spectral backscatter decomposition!**

\[ R_{aiwb}(t) = \int_0^x G_{aiwb}(z) T_{aiwb}(t-z) \, dz + \epsilon_{aiwb}(t) \]

\[ = \rho_{aibw} \int_0^x C_{aib}(z) T_{aiwb}(t-z) \, dz + \epsilon_{aiwb}(t) \]

\[ = \rho_{aibw} Q_{aib}(t) + \epsilon_{aiwb}(t) \]
The Lidar Data

• Thus, we can use alternating regressions to also estimate the spectral backscatter using the observed data, **without deconvolution**

• Using the observed data or using the deconvolved data still results in estimates of the spectral backscatter

• Reasonable conjecture that the observed data will do better?
Frequency Agile Lidar Data

Observed data

Backscatter averaged across bursts

Almost perfect classification if the first 2 waves are used.
Deconvolution

Backscatter averaged across bursts
The Lidar Data

• The leave-one-out error rate when using the observed data = 4%

• The leave-one-out error rate when using the deconvolved data = 11%
• Within a sample, for every wavelength, we have functional data in burst and time

• As it turns out, the functional data have a structure that is generally well modeled by a 3-parameter PDE

• Rewrite

\[ R_{aw}(t) = R_{iw}(b,t) \]
Then, a pretty good approximation for each wavelength is the solution to

\[
\frac{\partial R_{aiw}(b,t)}{\partial b} - \theta_1 \frac{\partial^2 R_{aiw}(b,t)}{\partial t^2} - \theta_2 \frac{\partial R_{aiw}(b,t)}{\partial t} - \theta_3 R_{aiw}(b,t) = 0
\]

This model came from PDE experts, who looked at the data and said “oh, it looks like the above”

Thus, each sample has 57 parameters: 19 wavelengths x 3 parameters
We pre-normalized so that the maximum signal for each wavelength = 1

We have developed new penalized-B-spline based approaches for computationally efficient estimation of the parameters, that also have standard errors
• Both methods use penalization

• The frequentist method uses what Ramsay and Cao call parameter cascading with data-based estimation of penalties

• The Bayesian method is more complex to formulate

• Both work much better than standard approaches without penalties
The dimension reduction using PDE is from 3,800,000 observations to 57.

The errors rates when applied to the actual data, and to the deconvolved data, are the same as using spectral backscatter.
Initial Summary

• In the LIDAR, anything you want to do with the deconvolved data can also be done with the observed data

• This is particularly true of the spectral backscatter operation

• In all cases, avoiding deconvolution lowers error rates
Initial Summary

• We have also done simulations where we simulated from a PDE model to replicate the received data.

• In all of them, deconvolution leads to higher error rates
• There is no theorem that says deconvolution is bad.

• There are LiDAR-type settings where deconvolution can be beaten theoretically by using other methods.

• There are also other deconvolution-type settings where deconvolution can be beaten theoretically by using the observed data.
Another Option

\[ R_{ai}(t) = \int_0^x G_{ai}(z) T_{ai}(t-z) \, dz + \epsilon_{ai}(t) \]

- Then take means so that

\[ R_{ai}(t) = \mu_{a}(t) + \delta_{ai}(t) \]

- Evaluate the function on a grid

\( (t_1, \ldots, t_M) \)
Another Option

\[ R_{ai}(t) = \mu_a(t) + \delta_{ai}(t) \]

- Now collect the data for a sample into a vector

\[ \tilde{R}_{ai} = \tilde{\mu}_a + \tilde{\delta}_{ai} \]

- This starts to look like a regular multivariate classification problem
Another Option

\[ \tilde{R}_{ai} = \tilde{\mu}_a + \tilde{\delta}_{ai} \]

- The errors are distinctly correlated in these data

- Assume Gaussian errors. Then we can write this in vector form as

\[ \tilde{\delta}_{ai} = S \tilde{\xi}_{ai} \]

- Here \( \tilde{\xi}_{ai} \) is Gaussian white noise, so that

\[ \tilde{R}_{ai} = \tilde{\mu}_a + S \tilde{\xi}_{ai} \]
The Lidar Data

\[ \tilde{R}_{ai} = \tilde{\mu}_a + S\tilde{\xi}_{ai} \]

• The integral is meant to be a vector

• If we know \( S \), for example by fitting a time series model, then

\[ S^{-1}\tilde{R}_{ai} = S^{-1}\mu_a + \tilde{\xi}_{ai} = \eta_a + \tilde{\xi}_{ai} \]

• Fisher’s LDA, or a centroid classifier, is optimal for this problem
The Lidar Data

- In particular, this inversion will always beat deconvolution, since Fisher’s LDA, or a centroid classifier, is optimal for this problem
The Lidar Data

• The LiDAR data are a type of space (burst and time) time (wavelength) data

• As such, at least in principle, the same algorithm can be applied

• You simply need a low-dimensional approximation to the covariance matrix and a way to invert it.
The Lidar Data

- In the spectral backscatter world, the same is true
The Lidar Data

\[ S^{-1} \tilde{R}_{ai} = \eta_a + \tilde{\xi}_{ai} \]

- Our (Hall, Delaigle, Carroll) idea is to recognize that the data are a function evaluated on a grid.
- We then approximate \( S \) as if it were a well-behaved low-dimensional convolution operator, and then invert it.
- Simulations suggest this is a better approach than either full-blown deconvolution or using the data as is.
Another Setting

- The simplest statistical inverse problem I know about is measurement error modeling.
Measurement Error in Nonlinear Models
A Modern Perspective
Second Edition

Raymond J. Carroll
David Ruppert
Leonard A. Stefanski
Ciprian M. Crainiceanu
Measurement Error Regression

• I consider here a nonparametric measurement error model

• In this context, there is a true covariate $Z_{ijk}$ which over samples and classes has the same density $f_Z$
The Ideal World

- So far there has been no class effect

- In the ideal world, along with $Z$ we observe $Q_{ijk}$, where

$$Q_{ijk} = g_{ik}(Z_{ijk}) + \varepsilon_{ijk};$$
$$0 = E(\varepsilon_{ijk} | Z_{ijk})$$
The Real World

• We do not observe $Z$. Instead, we observe $W = Z + U$;

$$W_{ijk} = Z_{ijk} + U_{ijk};$$

$$U_{ijk} = f_U(\cdot) \text{ known}$$

• We then regress $Q$ on $W$, and our observed data are

$$Y_{ik,\text{obs}} = \left\{ \hat{g}_{ik,\text{obs}}(t_1), \ldots, \hat{g}_{ik,\text{obs}}(t_m) \right\}$$
The Real World

• Of course, we can solve the inverse problem (measurement error regression), to get

\[ \hat{X}_{ik,\text{decon}} = \left\{ \hat{g}_{ik,\text{decon}}(t_1), \ldots, \hat{g}_{ik,\text{decon}}(t_m) \right\} \]

• Which should we use for classification?

\[ \hat{X}_{ik,\text{decon}} \text{ or } Y_{ik,\text{obs}} \]
• Of course, if the inverse solution were of interest, we would solve it.

• There is a massive literature on how to do this.

• However, this is a classification problem! We are not trying to describe the structure of the data.
Strategies

• I will consider two methods for classification

• **Method 1**: Nonparametrically regress Q on $W$ in each sample, compute the fitted functions on a fixed grid, and call this the data for classification

• **Method 2**: Nonparametrically deconvolve to estimate the regression of Q on $Z$ in each sample, compute the fitted functions on a fixed grid, and call this the data for classification
I am not going into the details of kernel method theory.

Regular kernel/spline regression is well-established and many packages exist for it.

Deconvolution kernel regression is harder and less established: I have my own Matlab code for it.
Regular Kernel Regression

• The regression of Q on W can be optimally estimated nonparametrically at the rate $n^{-2/5}$

• That is, $\hat{g}_{ik,obs}(w)$ converges to the regression of Q on W at the optimal rate of $n^{-2/5}$

• There are plenty of R programs using generalized crossvalidation to estimate the bandwidth and obtain this optimal rate
Deconvolution Kernel Regression

• The regression of Y on W can be optimally estimated nonparametrically at the rate $n^{-2/5}$.

• If the density of U is Gaussian, the optimal rate for estimating the regression of Y on X is $\log(n)$!

• Basically, to get consistency, the variance of the estimate becomes enormous.

• Deconvolution adds infinite noise in this case.
Measurement Error Regression

• The theory is done for local cases where the regression functions in the two classes differ by a factor of $n-\frac{2}{5}$

• **Theorem**: No matter what the distribution of the noise is, *asymptotically deconvolution results in higher misclassification rates* than ignoring the measurement error.
I have done extensive simulations with estimated bandwidths and modest sample sizes, and *never have I found a case where deconvolution helps*.
Beating Deconvolution in LiDAR

\[ R_{aiwb}(t) = \int_0^x G_{aiwb}(z)T_{aiwb}(t-z) \, dz + \varepsilon_{aiwb}(t) \]

- Fix one wavelength and one burst for simplicity

\[ R_{ai}(t) = \int_0^x G_{ai}(z)T_{ai}(t-z) \, dz + \varepsilon_{ai}(t) \]

- Take its mean across samples

\[ \mu_a(t) \]
Conclusions

• Intuition suggests that deconvolving signals and then trying to classify might suffer compared to using the raw data

• This is because the deconvolution adds noise, sometimes a lot of noise

• I showed this more or less precisely in some simple contexts