Journal of Biomolecular NMR (2006) 34: 179–195
DOI 10.1007/s10858-006-0020-z

Article

PR-CALC: A program for the reconstruction of NMR spectra from projections

Brian E. Coggins & Pei Zhou*
Department of Biochemistry, Duke University Medical Center, Durham, NC 27710, USA

Received 31 October 2005; Accepted 23 January 2006

Key words: fast multidimensional NMR, PR-CALC, PR-NMR, processing software, projection-reconstruction NMR

Abstract

Projection-reconstruction NMR (PR-NMR) has attracted growing attention as a method for collecting multidimensional NMR data rapidly. The PR-NMR procedure involves measuring lower-dimensional projections of a higher-dimensional spectrum, which are then used for the mathematical reconstruction of the full spectrum. We describe here the program PR-CALC, for the reconstruction of NMR spectra from projection data. This program implements a number of reconstruction algorithms, highly optimized to achieve maximal performance, and manages the reconstruction process automatically, producing either full spectra or subsets, such as regions or slices, as requested. The ability to obtain subsets allows large spectra to be analyzed by reconstructing and examining only those subsets containing peaks, offering considerable savings in processing time and storage space. PR-CALC is straightforward to use, and integrates directly into the conventional pipeline for data processing and analysis. It was written in standard C++ and should run on any platform. The organization is flexible, and permits easy extension of capabilities, as well as reuse in new software. PR-CALC should facilitate the widespread utilization of PR-NMR in biomedical research.

Introduction

In recent years, significant efforts have been devoted to the development of fast methods of acquiring multidimensional NMR data. The conventional technique for measuring an *n*-D spectrum involves sampling an *n*-D time domain along a regular *n*-D grid, followed by an *n*-D Fourier transform. The sampling requirements for this method increase by orders of magnitude as more dimensions are added, such that significant instrument time is needed even for 3-D and low-resolution 4-D experiments. The required spectrometer time becomes prohibitively expensive for experiments with more than four dimensions. However, new methods that do not rely upon the complete, systematic sampling of the *n*-D time domain – including the reduced dimensionality, GFT, projection-reconstruction, multidimensional decomposition, filter diagonalization, nonlinear sampling (followed by maximum entropy reconstruction), Hadamard and single-scan multidimensional NMR approaches – offer the prospect of obtaining equivalent information in significantly less measurement time (Szyperski et al., 1993a, b; Schmieder et al., 1994; Brutscher et al., 1995; Hu et al., 2000; Mandelshtam, 2000; Hoch and Stern, 2001; Orekhov et al., 2001; Ding and Gronenborn, 2002; Frydman et al., 2002; Freeman and Kupče, 2003, 2004; Kim and Szyperski, 2003; Kupče and Freeman, 2003a; Malmodin and Billeter, 2005a).

*To whom Correspondence should be addressed.
E-mail: peizhou@biochem.duke.edu

Projection-reconstruction NMR (PR-NMR), which was recently introduced by Kupče and Freeman, is one of these techniques permitting rapid data collection (Kupče and Freeman, 2003b, c; Coggins et al., 2004, 2005; Kupče and Freeman, 2004a–c, 2005; Hiller et al., 2005; Jiang et al., 2005; Malmodin and Billeter, 2005b, c; Venters et al., 2005). PR-NMR involves measuring a series of lower-dimensional (in many cases, 2-D) projections of a higher-dimensional spectrum, taken at different projection angles, which are then used to reconstruct the full spectrum mathematically. The collection of projections is accomplished by incrementing the indirect evolution times simultaneously, at fixed ratios, such that each experiment measures a slice through the higher-dimensional time domain. According to the Fourier projection-slice theorem, the Fourier transform of such a slice gives the projection of the frequency domain, at the same angle as the slice (Bracewell, 1956, 2000; Nagayama et al., 1978). This mode of data collection shares in common with GFT that it is based on the "accordion" principle, with the simultaneous incrementing of evolution times, although the pattern of data sampling in GFT is different, and the method of interpretation in GFT does not involve reconstruction (Kim and Szyperski, 2003; Freeman and Kupče, 2004; Moseley et al., 2004; Atreya and Szyperski, 2005). In PR-NMR, the projection data serve as the input to the reconstruction process, which generates a full spectrum, identical in form to that determined by the conventional technique. The quality of the reconstruction depends upon the number of projections used, the resolution and sensitivity of the projections, the algorithm used for reconstruction as well as implementation details of the reconstruction software. We recently demonstrated that, with the appropriate sampling parameters and reconstruction algorithm, such as filtered backprojection (FBP), quantitatively accurate reconstructions can be achieved (Coggins et al., 2005). Additionally, a variety of studies have shown that reconstructions suitable for sequential backbone and sidechain assignment, which may lack quantitative peak volumes and lineshapes, but with highly accurate peak positions, can be obtained routinely from a small amount of data, using a variety of reconstruction methods, including the lower-value (LV), backprojection (BP) and hybrid BP/LV (HBLV) algorithms (Kupče and Freeman, 2003b, c, 2004a–c, 2005; Coggins et al., 2004; Jiang et al., 2005; Venters et al., 2005).

PR-NMR has been successfully demonstrated on a number of systems, of widely varying molecular weights, for several different purposes. Initial proof of concept examples included the rapid measurement of the 3-D HNCO spectrum of ubiquitin (Kupče and Freeman, 2003c), followed closely by the measurement of several other 3-D and 4-D PR-NMR backbone assignment spectra (Kupče and Freeman, 2004a, b), and the application of PR-NMR to determine the 5-D HACACONH spectrum of protein G B1 domain (Coggins et al., 2004). For more routine use in structural biology and biophysics research, we have subsequently developed a suite of (4,2)-D pulse sequences for backbone assignment in large proteins as well as (4,3)-D pulse sequences for sidechain assignment in small to medium size proteins (Jiang et al., 2005; Venters et al., 2005). The former have been demonstrated on proteins as large as 30 kDa, with reconstruction producing complete 4-D spectra with no artifacts, permitting complete backbone assignment (Venters et al., 2005). An exciting application, which would not have been possible without PR-NMR, was the sequential assignment of a small protein *in vivo* (Reardon and Spicer, 2005). Additionally, a recent study has applied PR to NOESY for the first time, determining the high resolution 4-D methyl/amide-NOESY spectrum of the 29 kDa protein human carbonic anhydrase II from 2-D projections collected in 3.6 days on an 800 MHz spectrometer with cryoprobe, the same amount of time needed for a conventionally-measured 3-D experiment (Coggins et al., 2005). The methods used permitted quantitative reconstruction, verified by peak integration.

At the heart of the PR-NMR process is the reconstruction of spectra from the projection data. We describe here the program PR-CALC, a complete package for PR-NMR reconstruction. PR-CALC implements most of the reconstruction algorithms described to date, and has been carefully optimized to obtain maximal accuracy and performance. The program can reconstruct full spectra of any size or dimensionality, but it can also reconstruct arbitrary subsets of spectra as desired, an advantage that can lead to considerable savings in processor time and storage space. PR-CALC was designed to integrate well with common NMR processing and analysis software, and it has a simple, straightforward user interface, making the reconstruction process seamless and automatic. Because it works with data in standard file formats, the program can be used for data measured on any type of spectrometer. PR-CALC is written in standard C++ and should run on any computing platform; it is designed to take advantage of multiprocessor systems and 64-bit memory addressing if available, and will be extended in the future to support cluster computing environments, but it also functions well on ordinary hardware. PR-CALC was used for many of the published demonstrations of PR-NMR, and as such, it has been thoroughly tested. In addition, PR-CALC was built so that it could be integrated directly with other software, providing "behind-the-scenes" data processing services, and many functions are available as a library for further software development. The PR-CALC program, coupled with standard processing and analysis software, permits the straightforward, routine use of the projection-reconstruction technique in biomolecular NMR.

Methods

Program architecture

PR-CALC was implemented in ISO standard C++, with a modular, hierarchal structure (Figure 1a). This organization separates the different functions of the program into layers, which are implemented as independent software modules, making it easy to extend the program, and to reuse portions for other purposes. C++ object hierarchies are employed extensively to give the program a logical, intuitive organization. Functions for accessing and storing NMR data are managed within the lowest level, which provides an abstract representation of NMR data, hiding all details of storage and file formats from higher levels of the software. Support is provided for the NMRPipe (Delaglio et al., 1995), NMRView (Johnson and Blevins, 1994), XEASY (Bartels et al., 1995) and UCSF/Sparky (Goddard and Kneller) data formats, as well as raw binary and text files, and the program is set up so that support for other file formats can be added easily. The reconstruction algorithms have been implemented in the middle layer of the program. A C++ object hierarchy provides representations for PR-NMR experiments, and the associated projections and reconstructions, as well as functions for executing calculations, and for managing and distributing reconstruction jobs. The reconstruction algorithm implementations, including the lower-value (LV) (Kupče and Freeman, 2003c; Coggins et al., 2004), backprojection (BP) (Kupče and Freeman, 2004c), hybrid backprojection/lower-value (HBLV) (Venters et al., 2005) and filtered backprojection (FBP) (Coggins et al., 2005) algorithms, have been optimized extensively to achieve maximal calculation speed, as well as accuracy. The highest layer of the program manages the user interface, as well as the interface for interacting with other software. The program is operated by the user from a text shell using command-line switches, or by other programs using commands passed over pipes.

NMRData library for data access

All functions relating to the storage of NMR data have been isolated into an independent library, called NMRData, which constitutes the lowest layer of the software. This library provides a unified interface to all data via C++ object polymorphism: a single, common base class specifies the interface, but more complicated derived classes are used to provide support for the individual file formats (Figure 1b, c). Software that uses the library can manipulate the data, regardless of the file format or storage location, via a generic pointer or reference to a base class, and the software automatically substitutes the appropriate code for the particular case (Figure 1c, top). The library is designed to function with datasets that are too large to contain in memory, using a paging/caching mechanism, as well as smaller datasets and also temporary datasets. The library interface was built to have natural C++ container and object semantics (Figure 1b). Reference and value objects are provided for working with hypercomplex data points of arbitrary dimensionality; Standard Template Library (STL)-compatible iterators are provided for applying standardized algorithms to data; subsetting objects are provided for accessing portions of data; and standard operators have been extended to support simple, logical operations on the data. The NMRData library is designed for easy reuse in any NMR application, not just PR-NMR reconstruction.

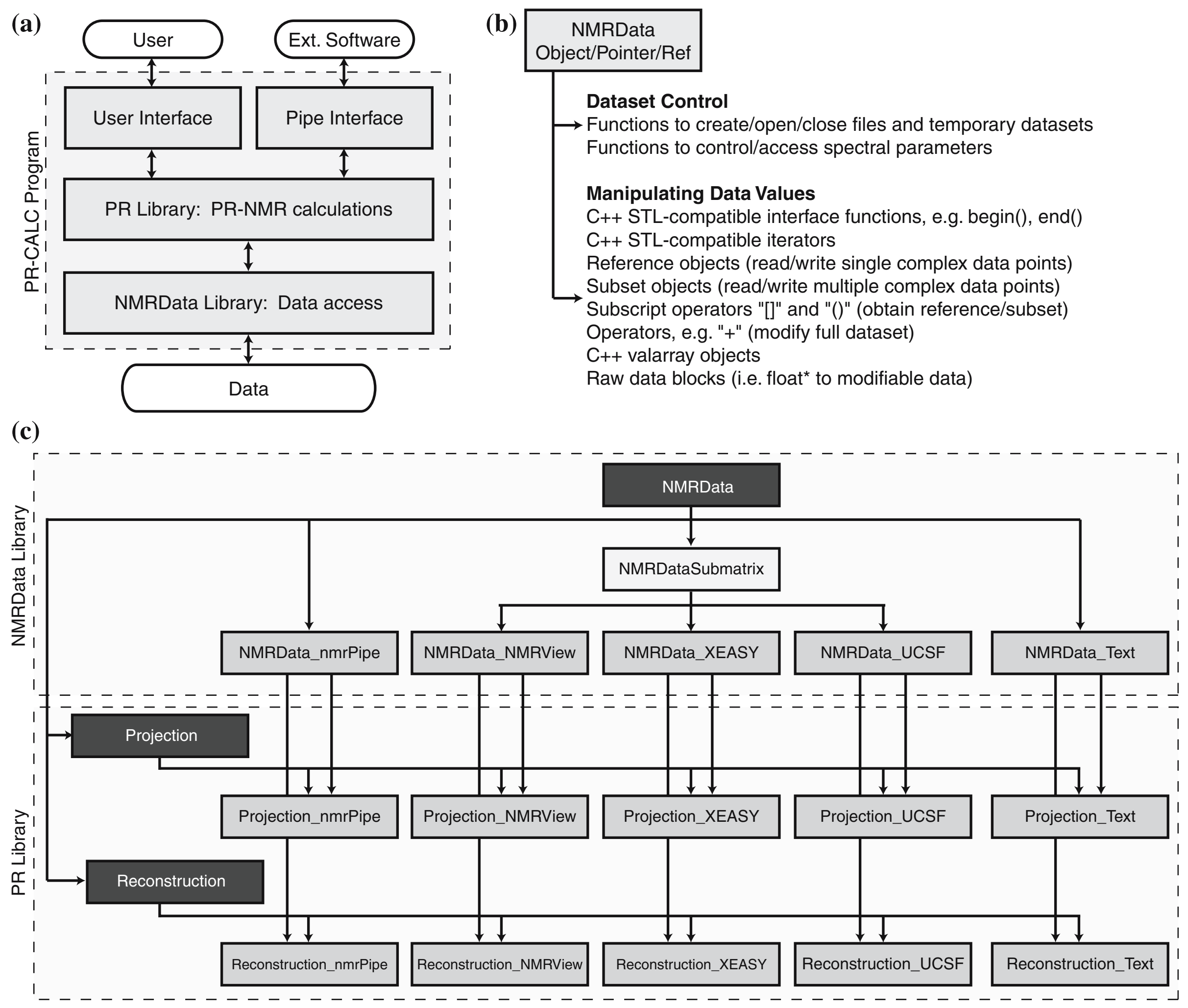

(a)
User
Ext. Software
PR-CALC Program
User Interface
Pipe Interface
PR Library: PR-NMR calculations
NMRData Library: Data access
Data
(b)
NMRData Object/Pointer/Ref
Dataset Control
Functions to create/open/close files and temporary datasets
Functions to control/access spectral parameters
Manipulating Data Values
C++ STL-compatible interface functions, e.g. begin(), end()
C++ STL-compatible iterators
Reference objects (read/write single complex data points)
Subset objects (read/write multiple complex data points)
Subscript operators "[]" and "()" (obtain reference/subset)
Operators, e.g. "+" (modify full dataset)
C++ valarray objects
Raw data blocks (i.e. float* to modifiable data)
(c)
NMRData Library
NMRData
NMRDataSubmatrix
NMRData_nmrPipe
NMRData_NMRView
NMRData_XEASY
NMRData_UCSF
NMRData_Text
PR Library
Projection
Projection_nmrPipe
Projection_NMRView
Projection_XEASY
Projection_UCSF
Projection_Text
Reconstruction
Reconstruction_nmrPipe
Reconstruction_NMRView
Reconstruction_XEASY
Reconstruction_UCSF
Reconstruction_Text

Figure 1. Internal Structure of the Software. (a) General organization of the PR-CALC software. (b) Summary of the standard software interface for accessing NMR spectral data, provided by the NMRData library. To access a dataset, a client can hold a generic pointer or reference to the NMRData base class, and use any of the interface methods listed. The same interface applies for all types of data storage. (c) Object hierarchy supporting multiple data file types, as well as projection and reconstruction interfaces, in the NMRData and PR libraries. Only a subset of the full library object hierarchy, pertaining directly to file type support, is shown. Classes in black are base classes defining interfaces, while classes in medium gray are the actual objects that are instantiated when the library is used; black arrows show inheritance. Pointers to the black base class types are used by client code; the derived types are generated within the library according to the C++ "factory" pattern. The complex multiple inheritance graph in the PR library allows the derived objects to support multiple interfaces.

PR library organization

The software representations of PR-NMR experiments and their projections and reconstructions, the program code for computing reconstructions from projections, and the tools for tracking and administering multiple reconstruction jobs are organized into a PR library. This library interacts with, and is built from, the NMRData library, which handles all data access on its behalf. Experiment objects are used to represent complete PR-NMR experiments, containing members that track the associated projection data, and that can compute reconstructions from that data. Projection objects are derived using multiple-inheritance from both a generic projection base class, as well as from the NMRData library objects (Figure 1c). This mechanism allows a projection object to be two things at once: It has an identity as a projection, including a relationship to the larger experiment and an interface for accessing that relationship; in addition, it is an NMR spectrum, and therefore shares the common interface used by the NMRData library for accessing data as well as managing referencing and spectral parameters. The same mechanism is used for reconstructions, which are derived by multiple-inheritance from a generic reconstruction base class, as well as NMRData classes for data management (Figure 1c). The library also provides a Job class, which stores an individual reconstruction request, and a Scheduler class, which coordinates the execution of reconstruction jobs using available processing resources.

Within this framework, the tasks that must be performed by software using the library to reconstruct a spectrum, and that are performed internally within PR-CALC, are straightforward. An Experiment object is created, which automatically loads its projections according to a description of the experiment and projection data given in an external ASCII text "control file" (see below). An empty Reconstruction object is created, and appropriate parameters defining the desired reconstruction are used to configure the object. A Job object is then created to specify how the task should be executed, and this is passed to the Scheduler. The Job object will track the progress of the calculation. After calculation is complete, the Reconstruction can be accessed in software like any other spectrum, using the common data interface.

Input data

PR-CALC accepts processed, frequency domain projection data, in any of the standard file formats named above, as input. Each projection is essentially an independent, lower-dimensional spectrum, and should be processed using standard processing approaches, as described in detail below. Projections may have any size and dimensionality, and there are no limits on the number of projections that may be used. The projections are permitted to have different dimensionalities, sizes and resolutions. For a successful reconstruction, however, the signal intensities should be consistent across the different projections. A scaling option can be used to rescale individual projections, when the signal levels vary.

General framework for reconstruction

Within the PR library, a common framework is employed for computing reconstructions using any of the supported algorithms, since the algorithms are all structured similarly, although the exact operations applied to the projection data differ between them. The following equation can be written to summarize the common process:

$$S(x,y,\ldots)=\mathop{\mathbf{X}}_{i=1}^{n} P_i(r),\ r=x\cos\alpha_{x,i}+y\cos\alpha_{y,i}+\cdots \tag{1}$$

where S represents the reconstruction, P_i represents the ith projection out of n, $\mathbf{X}$ represents the operations used for a particular algorithm, and $a_{u,i}$ represents the angle between reconstruction axis u and the projection axis for the ith projection. This equation considers only the indirect dimensions that participate in the projection-reconstruction process, but naturally, there would be an additional, directly-observed dimension for both S and P_i, and there may also be one or more indirect dimensions that are collected conventionally.

Equation 1 shows that each point in the reconstruction – denoted by S, a function of multiple Cartesian coordinates – can be determined independently of all other points in the reconstruction, through a set of operations performed on values from each projection. For a given reconstruction point in S, one value is read from each projection at the projection coordinate r, computed from the coordinates of the reconstruction point as well as the angles of the projection (i.e. x, y, $a_{x,i}$, $a_{y,i}$,...) (Coggins et al., 2004). Geometrically, this location is the intersection of the projection with a vector normal to the projection and passing through the point in S, a location which is termed the *point of projection* below. This geometry is shown in Figures 2a, b and 3a.

Note that Equation 1 treats each projection as if it were a continuous function of r. In reality, the projection values are known only for discrete positions of r, which will rarely coincide with the calculated points of projection; we must therefore use an interpolating scheme, working from the

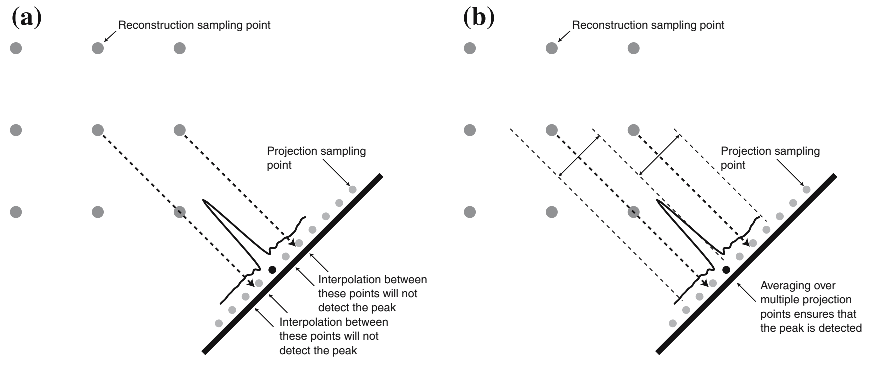

(a)
Reconstruction sampling point
Projection sampling point
Interpolation between these points will not detect the peak
Interpolation between these points will not detect the peak
(b)
Reconstruction sampling point
Projection sampling point
Averaging over multiple projection points ensures that the peak is detected

Figure 2. Interpolation vs. Averaging. These schematic diagrams compare the consequences of using interpolation *vis-à-vis* averaging in the determination of projection point values on projections, when the reconstruction grid resolution is significantly lower than the projection grid resolution. (a) Interpolation between projection points can fail to detect a peak, if the peak's linewidth is less than the spacing of reconstruction points. (b) Averaging will detect a peak no matter the difference in resolution.

known values. Thus the common operations needed for reconstruction are:

```
FOR EACH RECONSTRUCTION POINT:
1. FOR EACH PROJECTION:
   a. DETERMINE THE LOCATION OF THE POINT OF
      PROJECTION.
   b. INTERPOLATE A VALUE AT THAT POINT FROM
      KNOWN NEIGHBORING VALUES.
2. APPLY ALGORITHM-SPECIFIC
   OPERATIONS TO VALUES FROM
   ALL PROJECTIONS.
```

The interpolating method used in PR-CALC – an averaging algorithm – is described below. Although Equation 1 was written with the projection data as a function of a single tilted coordinate r, this was done only for clarity in the discussion. There is no reason that projections cannot have multiple tilted axes, as in the recent G^2FT experiments (Atreya et al., 2005). The implementation used in PR-CALC imposes no limitations on the number of tilted axes.

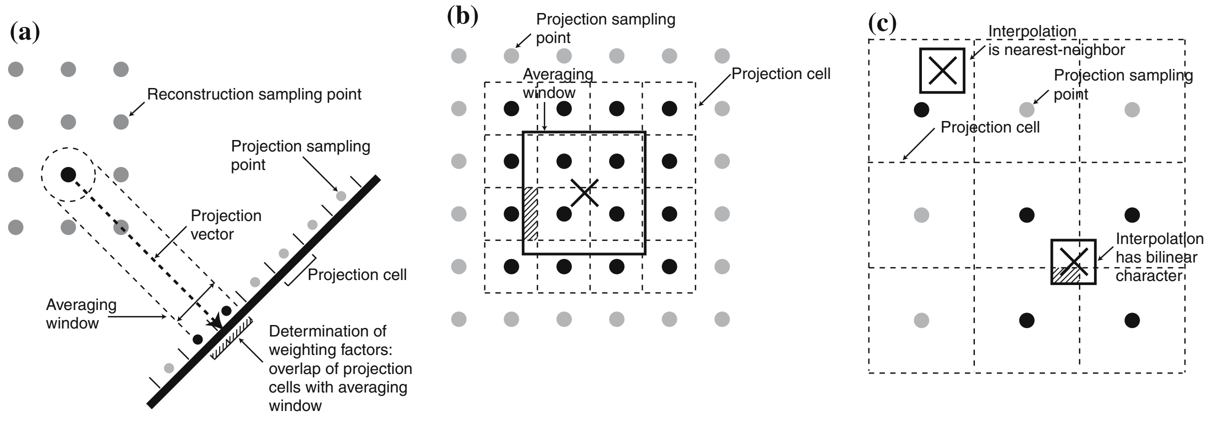

(a)
Reconstruction sampling point
Projection sampling point
Projection vector
Projection cell
Averaging window
Determination of weighting factors: overlap of projection cells with averaging window
(b)
Projection sampling point
Averaging window
Projection cell
(c)
Interpolation is nearest-neighbor
Projection sampling point
Projection cell
Interpolation has bilinear character

Figure 3. Averaging Algorithm for Projection Data. (a) Illustration of the averaging algorithm for a 2-D reconstruction grid, and a 1-D projection. The black reconstruction grid spot is being calculated. The averaging window size is determined based upon the spacing of reconstruction grid points. Each projection point is considered to have a "cell" surrounding it, as marked. The projection value is determined by averaging the values for all sampling points whose cells overlap with the averaging window (black sampling points). The relative contributions from each cell are determined by their overlap integrals with the averaging window (hatched), i.e. the hatched portion of the cell divided by the cell size gives the weighting factor. (b) Application to a 2-D projection. The point of projection is marked as a large "X", and the averaging window as a heavy box surrounding it. The dashed lines show the projection cells. Projection points in dark black contribute to the calculation of the projection value. The contribution of a single projection cell to the value is determined by the overlap integral; the weighting factor is calculated as the fraction of the cell's area that overlaps with the window. The overlap region is shown (hatched) for one cell; the weighting factor for this cell is therefore the area of the hatched region divided by the total area of the cell. (c) When the projection is sampled at lower resolution than the reconstruction, the averaging windows will be smaller than the size of the projection cells. In this case, the reconstruction can have either nearest-neighbor or linear characteristics.

Fundamentally, the reconstruction is executed as a loop over all points in the reconstruction, carrying out the above steps for each point. However, because of the number of iterations required, which can easily exceed 10^9 for 4-D spectra, it is essential that as many tasks as possible be removed from the inner loop. Memory allocations and other setup tasks are done prior to starting the loop, and all parameters needed for reconstruction are precalculated and reused throughout. Points of projection are determined by (1) precalculating the location of the point of projection onto each projection for the first point in the reconstruction, as well as (2) precalculating generic offsets for the next points in the reconstruction in each dimension. Thus, the inner loop can track the "current" points of projection by applying the offsets for each iteration, a simple vector addition rather than a complicated trigonometric computation. Note that the directly-observed dimension, as well as any indirect dimensions that were sampled conventionally, are treated identically to tilted dimensions in these procedures. The application of the averaging algorithm in these dimensions makes it possible to reconstruct at different resolutions in these dimensions than were used in the data collection.

Because each point in the reconstruction can be calculated independently, it is not necessary to calculate the entire spectrum in order to obtain a subset. PR-CALC can therefore be configured to calculate slices or small regions of experiments, as well as full experiments. The main calculation loop is configured automatically based upon the parameters of the reconstruction request.

Averaging of projection point values

Because the projection data are sampled at discrete positions, and because these positions generally do not coincide with projection vectors from reconstruction grid points, it is necessary to employ an interpolating method when accessing projection values. Although the nearest-neighbor and linear interpolation methods are the most obvious possibilities, our experience has shown them to be insufficient for effective PR-NMR reconstruction. Especially problematic are cases where the resolution of the reconstruction is significantly less than that of the projections: In this scenario, either of those methods can easily miss a peak that falls between sampling points (Figure 2).

Instead of an interpolation algorithm, we have employed an algorithm that averages the values from multiple sampling points on the projection. When reconstructing at low resolution relative to the projections, the averaging algorithm has the effect of smoothly downsampling the projections, preserving peak volume but decreasing resolution. The reconstruction will thus contain all peaks, broadened as if collected by conventional methods at low resolution. When reconstructing at the same resolution as the projections, this averaging algorithm behaves as a simple interpolation method. Note that in almost all cases, the digital resolutions of the various projections in an experiment will be different, since the spectral widths are adjusted to prevent aliasing on the projections. The averaging algorithm handles this situation automatically. The variation in digital resolution has not produced any noticeable negative effect on the reconstruction results to date, but the exact consequences, especially for quantitative FBP reconstructions, are under further investigation.

Before the reconstruction begins, the size of the averaging window is determined for each dimension of each projection. The averaging window is effectively the linear distance/area/volume (depending upon the dimensionality) of the projection that will be averaged to determine a projection value. When the reconstruction points are spaced equally in all dimensions, the averaging window size is set to the distance between reconstruction points on the reconstruction axes (Figure 3a). The situation is more complicated when the spacing of reconstruction points varies between dimensions. In this case, the empirically most successful solution is to determine the averaging window size for an individual projection as the linear combination of reconstruction point spacings in the reconstruction dimensions, weighted by the projection angles:

$$w = \sum_{i=1}^{n} \frac{sw_i \cos \alpha_i}{\mathrm{size}_i} \qquad (2)$$

where w is the window size in Hz, n is the number of reconstruction dimensions, sw_i is the spectral width of reconstruction axis i in Hz, α_i is the angle between the projection axis and axis i and size_i is the number of points on reconstruction axis i.

The averaging algorithm itself functions as follows. Each point on the projection is treated as having a rectangular "cell" extending in each direction to half the distance between points. A sum is accumulated as the algorithm progresses:

1. CALCULATE THE LOCATION OF THE POINT OF PROJECTION.
2. OVERLAY THE AVERAGING WINDOW, CENTERED ON THE POINT OF PROJECTION.
3. DETERMINE WHICH SAMPLING POINTS OF THE PROJECTION ARE INSIDE THE WINDOW.
4. FOR EACH POINT INSIDE THE WINDOW, AS WELL AS THE NEXT NEIGHBORING POINTS IMMEDIATELY OUTSIDE:
   a. DETERMINE A WEIGHTING FACTOR BASED UPON THE PORTION OF EACH POINT'S CELL INSIDE THE WINDOW.
   b. ADD THE PRODUCT OF THE VALUE AT THE SAMPLING POINT AND THE WEIGHTING FACTOR TO AN ACCUMULATING SUM.
5. DIVIDE THE ACCUMULATED SUM FOR ALL POINTS BY THE LENGTH/AREA/VOLUME OF THE WINDOW.

This is illustrated in Figure 3a for 1-D projections, and in Figure 3b for 2-D projections. The effect of the algorithm is to average all the points on the projection that lie inside the averaging window, with a fractional weighting for points that are only partly inside the window. When applied in a situation of equal or higher resolution in the reconstruction than in the projection, the averaging window will be smaller than the projection cells, causing it to function as an interpolation algorithm. Depending upon the specific circumstances, this interpolation can have nearest-neighbor or linear characteristics (Figure 3c).

Implementation of lower-value reconstruction

The LV reconstruction procedure is quite straightforward (Kupče and Freeman, 2003c; Coggins et al., 2004). Using the nomenclature of Equation 1, it is written as:

$$S(x, y, \ldots) = \operatorname{sgn}[P_i(r)] \min_{i=1}^{n} |P_i(r)|, \qquad r = x\cos\alpha_{x,i} + y\cos\alpha_{y,i} + \cdots \tag{3}$$

where sgn(x) is the sign or signum function:

$$\operatorname{sgn}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1 & x > 0 \end{cases} \tag{4}$$

Thus the projection value with absolute value closest to zero is assigned to the reconstruction point. Although the absolute value is used for the comparison, the actual value assigned has the original sign.

Implementation of backprojection reconstruction

The BP reconstruction procedure differs only slightly from LV (Kupče and Freeman, 2004c):

$$S(x, y, \ldots) = \frac{1}{n}\sum_{i=1}^{n} P_i(r), \qquad r = x\cos\alpha_{x,i} + y\cos\alpha_{y,i} + \cdots \tag{5}$$

The values from the projections are summed, and normalized by the number of projections.

Implementation of the hybrid backprojection/lower-value algorithm

HBLV uses a backprojection step, in which BP results are computed for different combinations of projections, followed by a lower-value comparison between these BP results (Venters et al., 2005). In the original formulation of the method, all possible combinations of k projections, where k is specified by the user, are tested. This can be written as:

$$S(x, y, \ldots) = \operatorname{sgn}\left[\sum_{P_j\in A_i}^{k} P_j(r)\right] \min_{i=1}^{{}_nC_k} \left|\sum_{P_j\in A_i}^{k} P_j(r)\right|, \qquad r = x\cos\alpha_{x,i} + y\cos\alpha_{y,i} + \cdots \tag{6}$$

In this equation, there are ${}_nC_k$ possible combinations of projections. Each possible combination is denoted as A_i. The k projections belonging to A_i (i.e.,$P_j \in A_i$) are summed, and the absolute values of the results for each combination are then compared, taking the lowest value. In practice, the sums are computed sequentially, and the software tracks the running lowest-value, rather than computing and storing sums for all combinations before a comparison.

As described in the original publication of the HBLV algorithm, a modification can be made to this procedure that can both (1) make the noise level easier to interpret and (2) reduce the computation time significantly (Venters et al., 2005). If $\sum_{P_j\in A_i} P_j(r)$ for any combination A_i is below the noise level, the final reconstruction value also must be below the noise level. The noise level in the reconstruction can be estimated theoretically as $N\sqrt{k}$, where N is the average noise level on the projections. The modification is thus to compare each summation $\sum_{P_j \in A_i} P_j(r)$ to the predicted noise level, and to stop computing summations if a combination is seen that is at or below the estimated noise level. By default, PR-CALC will estimate the noise level on each projection, compute the average, and execute the modified procedure. However, the user may provide a value for N, which would then be used to determine $N\sqrt{k}$ for the comparison, or may disable this feature entirely, in which case all combinations are computed and tested. Examples of reconstructions with and without this feature are given in the Results section.

Implementation of filtered backprojection

The FBP reconstruction procedure is identical to BP (Equation 5), except that the projections must first be filtered (Coggins et al., 2005). The filtering operation modifies the lineshape of the peak and is equivalent to the application of a specific window function in the time domain (Figure 4). For each data vector of each tilted projection dimension, PR-CALC transforms the data into the time domain, applies the filter, and transforms back into the frequency domain. The Fourier transforms are carried out via the well-established FFTPACK library developed at the National Center for Atmospheric Research, using real-to-Hermitian-complex routines (Swarztrauber, 1982). The filter function is applied independently to both the real and the imaginary components.

By default, PR-CALC applies the theoretically most accurate filter function for the number of coevolved indirect dimensions (Radon, 1917; Bracewell and Riddle, 1967; Rowland, 1979; Shepp, 1980; Deans, 1983; Kak and Slaney, 1999; Coggins et al., 2005). For (u, v)-D experiments, with only a single tilted projection axis, this would be:

$$w(t) = t^{u-v} \tag{7}$$

where the coordinate t is the time domain coordinate. Thus for (3,2)-D and (4,3)-D experiments, $w(t) = t$, and for (4,2)-D experiments, $w(t) = t^2$. The (3,2)-D and (4,2)-D time domain functions, and their effects on peak lineshape, are shown in Figure 4. The justification for Equation 7 arises from the basic principles of FBP. The specific purpose of the filter function is to correct for oversampling of the time domain close to the origin in the radial sampling procedure. In a 2-D space sampled radially, the density of sampling points decreases inversely with the distance from the origin ($1/t$), which must be corrected by reweighting the intensities of points using the function t. Adding a dimension causes the sampling density to decrease according to $1/t^2$ instead of $1/t$, and the exponent of the filtering function must therefore be increased. Equation 7 arises by extension of this principle to higher dimensional spaces.

Output data

The final spectra produced by PR-CALC are stored in the user's choice of any of the file formats listed above. Reconstructions automatically include referencing information calculated according to the information given in the control file.

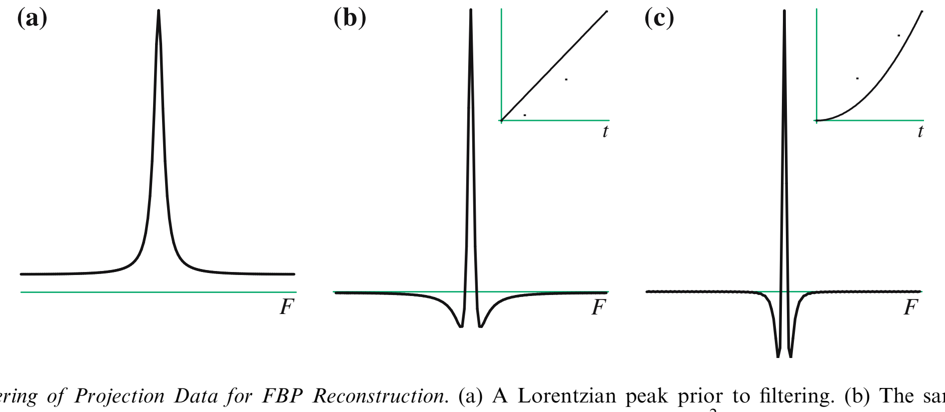


Figure 4. Filtering of Projection Data for FBP Reconstruction. (a) A Lorentzian peak prior to filtering. (b) The same peak, after filtering with the $w(t) = t$ function (inset) for (3,2)-D data. (c) After filtering with the $w(t) = t^2$ function (inset) for (4,2)-D data.

User interface

The program uses a simple command-line interface. Reconstruction requests are provided by options entered on the command-line, or alternatively as text files with multiple reconstruction commands, which are executed as a batch. The batch mechanism is an efficient way to compute a number of reconstructions from the same input data, as the overhead of loading projection data – and for FBP, filtering the data – need be done only once. A progress report display is provided, showing the status of running and queued calculations, and allowing calculations to be stopped in place and restarted later.

Pipe-based interface

A pipe-based interface is also provided, so that PR-CALC can be run in the background from other programs, as a data-processing server. Reconstruction commands are passed via a pipe, and PR-CALC provides status notifications in return.

Parallel computing

PR-CALC currently provides support for parallel computing on symmetric multiprocessing (SMP) systems. Reconstruction requests, received either from the command-line options, a text command file for batch reconstruction, or via a pipe, are queued and distributed among available processors automatically. To facilitate this feature, PR-CALC must be compiled with the Boost C++ threading library. Future plans call for extending PR-CALC to support parallel processing on multinode clusters.

PRSP

The utility program PRSP is included with PR-CALC. PRSP acts on Varian spectrometer data to separate intermodulated projections for (3,2)-D, (4,3)-D, (4,2)-D, (5,3)-D and (5,2)-D projection data. The program is run from the command-line, with the user supplying the dimensionality of the data; all other required parameters are read from the file header. PRSP reads a single unseparated projection data file and produces a set of independent projection files in Varian format, which are suitable for direct input into NMRPipe for processing. The details of intermodulated projections in PR-NMR have been described elsewhere (Kim and Szyperski, 2003; Kupče and Freeman, 2003c; Freeman and Kupče, 2004). An example for (3,2)-D data is given in the Results section below.

Compiling/system requirements

PR-CALC can be compiled on any system with an ISO standard C++ compiler and all ISO standard C++ libraries present. In addition, a FORTRAN compiler is required in order to compile the FFTPACK library, which is used for Fourier transforms. The GCC/G++/G77 compiler system, available on most platforms, is entirely sufficient to build the package. If the Boost C++ libraries are available, PR-CALC can be compiled with support for threading. The program has been tested extensively on recent versions of Linux, Microsoft Windows and Apple Mac OS X. PR-CALC compiles and runs without problem on 64-bit systems, and will take advantage of 64-bit capabilities, when applicable.

Availability

The full PR-CALC package, including source code, documentation and example data, is available online from http://zhoulab.biochem.duke.edu/software/pr-calc/. In addition, distributions with precompiled binaries for some platforms are available.

Results and discussion

PR-CALC has been used in a number of published PR-NMR studies, including the first reconstruction of a 5-D spectrum from projections (Coggins et al., 2004), the demonstration of (4,2)-D PR-NMR experiments for sequential assignment on 29 and 30 kDa proteins (Venters et al., 2005), the demonstration of sidechain assignment in small proteins using (4,3)-D experiments (Jiang et al., 2005), the first *in vivo* sequential assignment (Reardon and Spicer, 2005), and the first reconstruction of a NOESY spectrum from projections (Coggins et al., 2005). The program has been thoroughly tested, and has proven itself by the routine production of accurate spectra, as shown in these prior studies. We provide here more detailed information on how PR-CALC is used in practice to achieve these results. We also provide specific comparisons of results with several different calculation options, which have not been discussed extensively to date.

Approach to collecting and processing PR-NMR data

Our software for PR-NMR data processing is designed to integrate directly into the conventional data processing and analysis pipeline (Hoch and Stern, 1996), so that PR-NMR can be used with an approach only slightly different from the traditional one (Figure 5).

Data collection involves pulse sequences essentially identical to their conventional counterparts (i.e. magnetization transfer steps, and the means of achieving them, are identical), except that PR-NMR sequences must be modified to increment evolution times simultaneously in order to achieve a tilted projection (Kupče and Freeman, 2003a, 2004a, b; Coggins et al., 2004; Freeman and Kupče, 2004). In addition, minor modifications may be needed to ensure that all indirect dimensions have identical phase corrections, and to ensure that no distortions occur for specific tilt angles, as described by Venters et al. (2005), and the spectral widths of the projections must be adjusted as described previously to ensure that signals do not become aliased (Coggins et al., 2004). Typically, a given PR-NMR pulse sequence is run a number of times with different angle parameters, producing a collection of lower-dimensional time domain data files, each containing a projection at unique angles.

From the spectrometer to the final reconstructed spectrum requires only a few processing steps, described in detail below. Intermodulated projections must be separated first. The individual projections are then processed using standard NMR software and procedures, to give frequency domain projection data. These data are then the input for PR-CALC, which produces a final spectrum suitable for analysis. The only new software components required are PR-CALC and a tool to separate intermodulated projections; the included PRSP tool (*PR*/*S*eparate *P*rojections) meets the latter need for Varian spectrometer data.

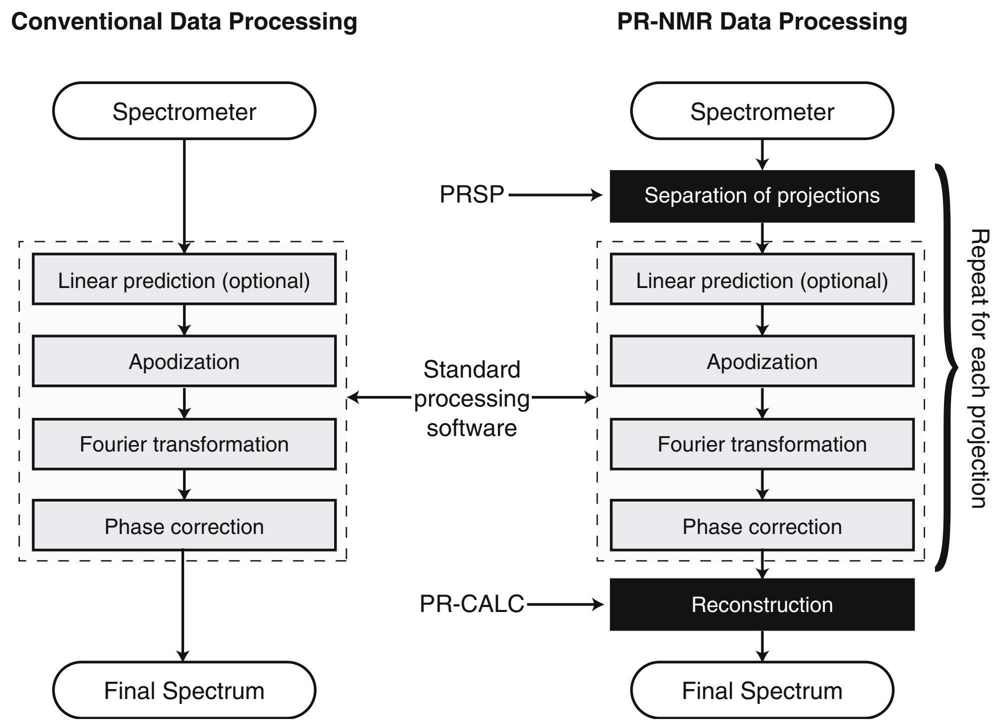

Conventional Data Processing
PR-NMR Data Processing
Spectrometer
Linear prediction (optional)
Apodization
Fourier transformation
Phase correction
Final Spectrum
Standard processing software
PRSP
Separation of projections
Repeat for each projection
PR-CALC
Reconstruction

Figure 5. Comparison of Data Processing Procedures for PR-NMR vs. Conventional Data. Data processing steps that are unique to PR-NMR are shown in black.

Separating intermodulated projections

The fact that the time domain is hypercomplex introduces a peculiarity to the measurement of projections: Measuring a hypercomplex slice at positive angles through the time domain provides enough information to describe frequency domain projections at negative angles, as well as positive angles. This phenomenon has been termed *intermodulation*, and its mathematical details, deduced using simple trigonometry, have been described in several publications (Kim and Szyperski, 2003; Kupče and Freeman, 2003c; Freeman and Kupče, 2004). For a tilted axis formed by coevolving *d* dimensions, with quadrature detection in all *d* dimensions, one obtains $2^{d\text{-}1}$ projections at different combinations of positive and negative angles for each projection experiment.

The first step of PR-NMR data processing is to separate these intermodulated projections. The time domain data cannot be Fourier transformed until this step is performed. The separation calculation applies simple trigonometric relationships to the data points to produce independent time domain datasets with the correct number of hypercomplex components. These datasets can thereafter be treated as independent projections at different angles.

For data recorded on Varian spectrometers, the PRSP tool included in this package can carry out this process automatically. It separates projections based upon the dimensionality, decoding gradient-based sensitivity-enhancement first if necessary (Kay et al., 1992). The FIDs in projections are expected to have a specific organization, and the resulting separated files are labeled according to the signs of the angles. For example, for a (3,2)-D experiment, PRSP expects the following FIDs, in the order given:

$$\begin{aligned}
\text{FID1}&: (\cos\omega_1 t_1)(\cos\omega_2 t_2)\\
\text{FID2}&: (\cos\omega_1 t_1)(\sin\omega_2 t_2)\\
\text{FID3}&: (\sin\omega_1 t_1)(\cos\omega_2 t_2)\\
\text{FID4}&: (\sin\omega_1 t_1)(\sin\omega_2 t_2)
\end{aligned}\tag{8}$$

Assuming that a projection were collected at angles α_1 to axis 1 and α_2 to axis 2, and that the original data file were named **fid**, PRSP would produce two files, **fid**_++ and **fid**_−+, containing the projections with angles $(+\alpha_1, +\alpha_2)$ and $(-\alpha_1, +\alpha_2)$, respectively, according to the following linear combinations of the starting FIDs:

$$\begin{aligned}
&\text{Projection} ++\begin{cases}\cos(\omega_1 t_1+\omega_2 t_2)=\text{FID1}-\text{FID4}\\ \sin(\omega_1 t_1+\omega_2 t_2)=\text{FID2}+\text{FID3}\end{cases}\\
&\text{Projection} -+\begin{cases}\cos(-\omega_1 t_1+\omega_2 t_2)=\text{FID1}+\text{FID4}\\ \sin(-\omega_1 t_1+\omega_2 t_2)=\text{FID2}-\text{FID3}\end{cases}
\end{aligned}\tag{9}$$

Full details on PRSP requirements for data organization are given in its manual.

Processing projection data

Once intermodulated projections are separated, the independent projections can be processed using standard NMR data processing tools and techniques, like any other lower-dimensional spectra (Hoch and Stern, 1996). At a minimum, this must include apodization, Fourier transformation and phase correction. Linear prediction, and other procedures such as solvent artifact suppression, may be used if desired. Ideally, each of the projections is processed identically; this can be done conveniently with the aid of shell scripts.

Calculating reconstructions

Before the spectrum can be reconstructed, a text "control file" must first be created, describing the experiment and the projection data. An example control file, for a (4,2)-D methyl/amide NOESY experiment, is shown in Figure S1 in the Supplementary Information. During the preparation of the control file, it is important to control for variations in signal levels between the projections, by setting appropriate scaling factors. A detailed discussion of how to adjust signal levels is given in Appendix A (Supplementary Information).

Calculating reconstructions in PR-CALC is straightforward, once the control file has been constructed. A reconstruction is accomplished with a single command, such as:

```
pr-calc -in control.txt
-out spectrum.nv -lv
```

which instructs the program to load the experiment described in **control.txt**, compute a reconstruction of the full spectrum at the resolution specified in the control file, and to write the full spectrum to an NMRView file entitled **spectrum.nv**. The resulting spectrum is ready for analysis using NMR spectrum analysis programs. This reconstruction would use the LV algorithm, on account of the **–lv** option. To use HBLV with $k = 8$ would require:

```
pr-calc -in control.txt-out
spectrum.nv -hblv 8
```

The algorithm is selected using the **–lv**, **–bp**, **–hblv** and **–fbp** options.

To set the digital resolution of the reconstruction, the **–size** option is used, or the values in the control file are adjusted. An example of the former would be:

```
pr-calc -in control.txt-out spectrum.nv
    -size 64×64×64×512
```

which would produce a spectrum with 64 × 64 × 64 × 512 point digital resolution. The resolution may be set as desired, noting the caveats below, with PR-CALC downsampling or interpolating the projection data as needed using the averaging algorithm. Choosing a low reconstruction resolution will naturally reduce the quality of the reconstruction: All peaks will broaden, those that are close together will merge, positions will become inaccurate due to the low resolution sampling grid, and some peaks may broaden so much as to be lost to the noise. Depending upon the reconstruction algorithm, lowering the resolution may have other important consequences. For example, in a low resolution FBP reconstruction, the cancellation of ridges and troughs may not occur correctly, resulting in an increased apparent noise level. The best quality reconstructions are obtained if the reconstruction is carried out at a resolution equal to the resolution of the projection data. Figure 6 provides a demonstration of these phenomena, showing the FBP reconstruction of a specific plane from a methyl/amide NOESY experiment at different resolutions.

By default, PR-CALC will reconstruct the entire spectrum, but specific subsets of a spectrum can also be requested. As shown in Figure 7, the **–slice** option is used to obtain a lower-dimensional section out of the experiment, while the **–region** option produces a result with the same dimensionality as the experiment, but containing only a portion of it.

A batch processing mode is also available, in which multiple reconstructions can be requested from the same projection data (i.e. covering different regions, or using different algorithms), while only requiring the data to be opened once. For this feature, a text file is prepared listing the desired reconstructions in a syntax that is essentially identical to the command-line syntax. This "command file" is invoked with **–cmd**:

```
pr-calc -in control.txt -cmd command.txt
```

HBLV noise option

The HBLV implementation provides the choice of comparing all possible combinations of k projections, or stopping when a combination is found that sums to a value below a noise floor (Venters et al., 2005). The latter option is recommended, and is the program default; the **–noise** option overrides. An example comparing reconstructions with both choices, for the reconstruction of a plane from an HNCACB spectrum of HCA II, is shown in Figure 8. The default setting leads to a noise level that more accurately reflects the true sensitivity of the experiment, and avoids the production of "spikes" that may be misinterpreted as peaks.

Performance

The processing time needed by PR-CALC scales with the number of data points to reconstruct, and the number of projections, according to their product:

$$T(N, n) = O(Nn) \qquad (10)$$

where T is the computation time, N is the number of data points in the reconstruction, n is the number of projections, and $O(x)$ indicates that the form of the asymptotic upper bound running time is x. Although every effort has been made to optimize the calculation steps of the inner loop, the aggregate time can become quite large for large N and n (e.g. a (4,2)-D FBP reconstruction can easily have $N = 10^9$ and $n = 10^2$, yielding 10^{11} iterations of the PR-CALC inner loop, which translates into several days of processing time on a 2.4 GHz Intel Xeon CPU). Distributing this burden among multiple processors introduces a slight overhead, but for many reconstructions this is insignificant compared

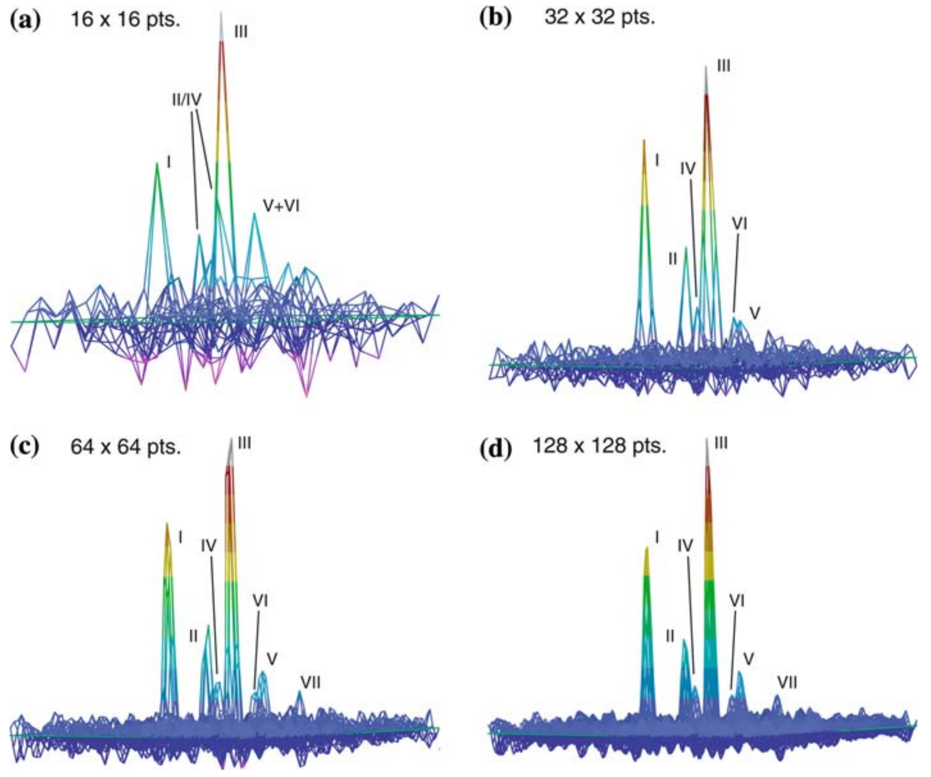


Figure 6. Reconstructions at Different Resolutions. The HM/CM plane at the HN and N coordinates for residue S50 (HN = 8.38 ppm, N = 122.55 ppm), from the (4,2)-D methyl/amide NOESY spectrum of human carbonic anhydrase II described previously (Coggins et al., 2005), is shown in stacked plots, reconstructed at different digital resolutions. The projections were recorded with 48 complex points in their tilted dimensions, and were subsequently extended by linear prediction to 64 points, and were zero-filled to 128 points. FBP was used for reconstruction. For all panels, the seven crosspeaks are: I, V78 H2; II, V49 H2; III, V49 H1; IV, V78 H1; V, L79 H1; VI, L44H1 → A257 HN, from an adjacent plane; VII, L79 H2. (a) 16 × 16 point digital resolution. Crosspeaks I and III are resolved, but broadened; the lack of resolution makes it impossible to resolve the remaining crosspeaks individually. The apparent noise level is high, reflecting lack of proper ridge and trough cancellation in the FBP process at low resolution. (b) 32 × 32 point digital resolution. All crosspeaks except VII, which is broadened into the noise, can be resolved individually. Peak widths are broader than the projection linewidths. The apparent noise level is lower than at 16 × 16 point resolution, but still above the natural noise level. (c) 64 × 64 point digital resolution. All peaks are resolved at the linewidths measured on the projections, and the apparent noise level matches the experimental noise level. (d) 128 × 128 point digital resolution. Crosspeak linewidth is identical to the 64 × 64 point reconstruction (i.e. in both cases, reconstruction linewidths match projection linewidths), but peak shapes are of higher quality, due to finer digital sampling. The extra resolution also improves the accuracy of peak positions.

to the actual calculation time. Naturally, the strategy of calculating only subsets from a large spectrum can reduce the calculation burden by many orders of magnitude, as computing a slice from a spectrum often requires no more than a few seconds with modern computers, and regions can be calculated on the order of minutes.

The exception to Equation 10 is the HBLV algorithm. If the noise floor is disabled, the processing time scales as:

$$T(N,n,k) = O[N \cdot {}_nC_k] = O\left(N\frac{n!}{k!(n-k)!}\right) \tag{11}$$

Needless to say, because of the factorials, for certain values of n and k, this time requirement can become prohibitively large. Besides having a large time requirement, this method also requires a large amount of system memory in order to track which projections are combined for each reconstruction bin. Because of these limitations, as a general rule, HBLV calculations for $n > 30$ are not possible. When the noise floor modification is active, the time requirement is reduced by orders of magnitude, but the exact reduction depends upon the data and therefore cannot be predicted *a priori*. Despite the acceleration provided by the noise

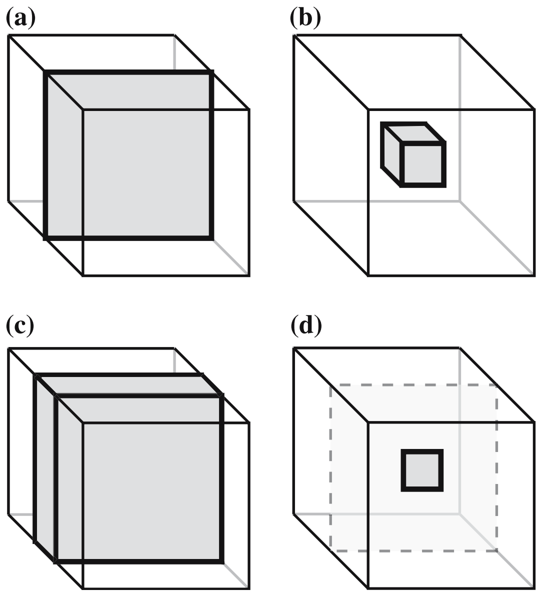


Figure 7. Reconstructing Subsets of Experiments. These schematic diagrams, for a 3-D experiment, show some of the possible subsets that can be built with PR-CALC. (a) The **–slice** option generates a lower-dimensional subset. In this case, a 2-D slice is shown. (b) The combination of **–center** and **–region** is used to obtain a subset with the same dimensionality as the full experiment, termed a "region." Here, a region is shown with reduced width in all three dimensions. (c) A region with full width in two dimensions, and a reduced width in the third dimension. (d) The slice and region mechanisms can be combined (using the three options **–slice**, **–center** and **–region**). In this case, a 2-D reconstruction is produced, containing a region out of a slice.

floor procedure, the maximal number of projections n will likely remain in the thirties, because of the explosive rate of growth of ${}_nC_k$ at those values and its consequences for calculation time and required memory.

Although Equations 10 and 11 may suggest significant limitations on the kinds of data that could be evaluated routinely by PR-NMR, we have found that with appropriate choices of strategy, the calculation burdens become quite manageable. A detailed discussion of these issues, including practical advice on how to approach large datasets, along with measured calculation times for certain common scenarios, is provided in Appendix B (Supplementary Information). In general, 3-D spectra can be reconstructed in full in only a few minutes, while 4-D spectra require hours to days, depending upon the input data and output resolution. A very useful practical approach for 4-D spectra is to reconstruct only the planes or regions containing peaks. Individual planes and regions can be calculated in seconds, and a complete set for all peaks in a spectrum can be obtained in no more than two hours in the worst case. For more details, please see the appendix.

Interaction with other software

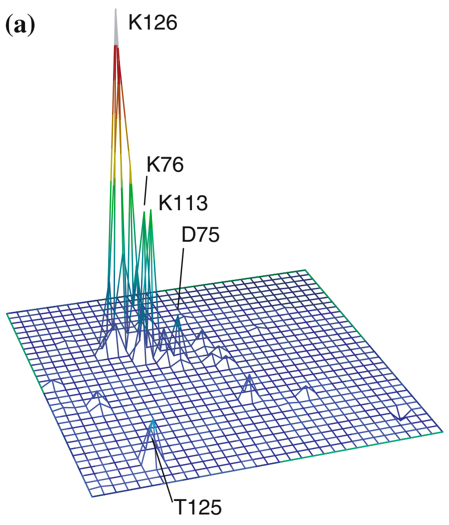
PR-CALC is designed to facilitate operation as a background service provider to other software. A special interface is activated using the **–cmdpipe** switch; in this mode, PR-CALC responds to formatted commands passed over a pipe from other software, with status updates returned over standard output. Reconstruction commands are dispatched via background threads.

This interface should prove particularly useful for connecting PR-CALC to spectrum analysis software that supports scripting. Custom, graphical interfaces could be built within analysis programs, making use of PR-CALC in pipe mode, allowing the automatic reconstruction of planes of interest from larger experiments, with simple controls for adjusting reconstruction parameters. Preliminary efforts to develop these capabilities for NMRView have proven successful and of great use in practice (D. Kojetin and J. Cavanagh, personal communication).

In addition, all of PR-CALC's capabilities are organized into C++ libraries, which could easily be used as components in the development of NMR software in C++. The library interfaces are described in detail in the program documentation, source code is provided, and licenses for library redistribution are available upon request.

Conclusions

Recent developments with projection-reconstruction NMR have shown it to be a method with broad applicability to problems in biomolecular NMR, allowing the collection of higher-dimensional experiments at very high resolution, with high quality, in no more time than would be needed to record lower-dimensional spectra conventionally. However, the widespread adoption of PR-NMR would not be possible without the



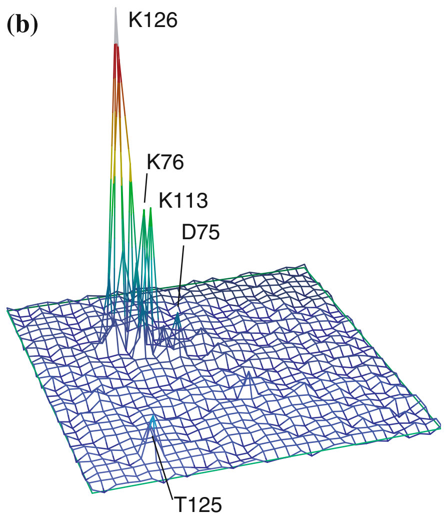


Figure 8. HBLV Noise Floor Option. Reconstructions of the CA/CB plane for residue K126 (HN = 7.965 ppm, N = 124.569 ppm) from the (4,2)-D HNCACB of human carbonic anhydrase II described previously (Venters et al., 2005), using HBLV with $k = 12$. (a) With the noise floor disabled. Note the artificially flat baseline, interrupted by small artifactural "spikes." (b) With the noise floor enabled. The reconstruction shows a noise level that is consistent with the theoretically-expected noise, and artifactural spikes are not visible above this noise level.

availability of appropriate software for computing reconstructions from projections. We have described a complete program for PR-NMR reconstruction, PR-CALC, that meets this need. This program integrates directly into the conventional procedures used for processing and analyzing NMR data, such that processing PR-NMR data requires only a small amount of additional effort beyond that needed to process conventionally-recorded data. It has been used successfully in many of the PR-NMR studies published to date, and has been shown to produce accurate spectra. PR-CALC has been optimized to achieve excellent performance, and to take advantage of the full capabilities of the available computing hardware. PR-CALC should meet the needs of anyone interested in applying PR-NMR to biological experiments, opening the door for the widespread utilization of this exciting new method.

Acknowledgements

We thank Dr. Ronald A. Venters for many helpful discussions, for supplying example and test data, for extensive help with testing this software, and for suggesting improvements throughout its development. We also thank Dr. Doug Kojetin, Dr. John Cavanagh, Dr. Geoffrey Mueller, Mr. Patrick Reardon, Dr. Ling Jiang, Dr. Ming Li, Dr. Ming-Tao Pai, Ms. Martha Bomar and Mr. Jon Werner-Allen for testing the software and providing helpful suggestions. This work was supported by the Whitehead Institute and by N.I.H. grant AI055588-01 to P.Z. B.E.C. is the recipient of an N.S.F. Graduate Research Fellowship.

References

Atreya, H.S., Eletsky, A. and Szyperski, T. (2005) *J. Am. Chem. Soc.*, **127**, 4554–4555.

Atreya, H.S. and Szyperski, T. (2005) *Methods Enzymol.*, **394**, 78–108.

Bartels, C., Xia, T.-H., Billeter, M., Güntert, P. and Wüthrich, K. (1995) *J. Biomol. NMR*, **5**, 1–10.

Bracewell, R.N. (1956) *Aust. J. Phys.*, **9**, 198–217.

Bracewell, R.N. (2000) *The Fourier Transform and Its Applications* McGraw-Hill, Boston.

Bracewell, R.N. and Riddle, A.C. (1967) *Astrophys. J.*, **150**, 427–434.

Brutscher, B., Cordier, F., Simorre, J.P., Caffrey, M.S. and Marion, D. (1995) *J. Biomol. NMR*, **5**, 202–206.

Coggins, B.E., Venters, R.A. and Zhou, P. (2004) *J. Am. Chem. Soc.*, **126**, 1000–1001.

Coggins, B.E., Venters, R.A. and Zhou, P. (2005) *J. Am. Chem. Soc.*, **127**, 11,562–11,563.

Deans, S.R. (1983) *The Radon Transform and Some of Its Applications* John Wiley & Sons, New York.

Delaglio, F., Grzesiek, S., Vuister, G.W., Zhu, G., Pfeifer, J. and Bax, A. (1995) *J. Biomol. NMR*, **6**, 277–293.

Ding, K. and Gronenborn, A.M. (2002) *J. Magn. Reson.*, **156**, 262–268.

Freeman, R. and Kupče, E. (2003) *J. Biomol. NMR*, **27**, 101–113.

Freeman, R. and Kupče, E. (2004) *Concept. Magnetic. Res.*, **23A**, 63–75.

Frydman, L., Lupulescu, A. and Scherf, T. (2002) *Proc. Natl. Acad. Sci. U.S.A.*, **99**, 15,859–15,862.

Goddard, T.D. and Kneller, D.G. SPARKY 3, University of California, San Fransisco.

Hiller, S., Fiorito, F., Wüthrich, K. and Wider, G. (2005) *Proc. Natl. Acad. Sci. U.S.A.*, **102**, 10,876–10,881.

Hoch, J.C. and Stern, A.S. (1996) *NMR Data Processing* Wiley-Liss, New York.

Hoch, J.C. and Stern, A.S. (2001) *Methods Enzymol.*, **338**, 159–178.

Hu, H., de Angelis, A.A., Mandelshtam, V.A. and Shaka, A.J. (2000) *J. Magn. Reson.*, **144**, 357–366.

Jiang, L., Coggins, B.E. and Zhou, P. (2005) *J. Magn. Reson.*, **175**, 170–176.

Johnson, B.A. and Blevins, R.A. (1994) *J. Biomol. NMR*, **4**, 603–614.

Kak, A.C. and Slaney, M. (1999) *Principles of Computerized Tomographic Imaging* IEEE Press, New York.

Kay, L.E., Keifer, P. and Saarinen, T. (1992) *J. Am. Chem. Soc.*, **114**, 10,663–10,665.

Kim, S. and Szyperski, T. (2003) *J. Am. Chem. Soc.*, **125**, 1385–1393.

Kupče, E. and Freeman, R. (2003a) *J. Magn. Reson.*, **162**, 300–310.

Kupče, E. and Freeman, R. (2003b) *J. Am. Chem. Soc.*, **125**, 13,958–13,959.

Kupče, E. and Freeman, R. (2003c) *J. Biomol. NMR*, **27**, 383–387.

Kupče, E. and Freeman, R. (2004a) *J. Biomol. NMR*, **28**, 391–395.

Kupče, E. and Freeman, R. (2004b) *J. Am. Chem. Soc.*, **126**, 6429–6440.

Kupče, E. and Freeman, R. (2004c) *Concepts Magn. Reson.*, **22A**, 4–11.

Kupče, E. and Freeman, R. (2005) *J. Magn. Reson.*, **173**, 317–321.

Malmodin, D. and Billeter, M. (2005a) *Prog. Nucl. Magn. Reson. Spectrosc.*, **46**, 109–129.

Malmodin, D. and Billeter, M. (2005b) *J. Am. Chem. Soc.*, **127**, 13,486–13,487.

Malmodin, D. and Billeter, M. (2005c) *J. Magn. Reson.*, **176**, 47–53.

Mandelshtam, V.A. (2000) *J. Magn. Reson.*, **144**, 343–356.

Moseley, H.N.B., Riaz, N., Aramini, J.M., Szyperski, T. and Montelione, G.T. (2004) *J. Magn. Reson.*, **170**, 263–277.

Nagayama, K., Bachmann, P., Wüthrich, K. and Ernst, R.R. (1978) *J. Magn. Reson.*, **31**, 133–148.

Orekhov, V.Y., Ibraghimov, I.V. and Billeter, M. (2001) *J. Biomol. NMR*, **20**, 49–60.

Radon, J. (1917) *Berichte der Sächsischen Gesellschaft der Wissenschaften Leipzig, Math-Phys. Kl.,*, **69**, 262–277.

Reardon, P.N. and Spicer, L.D. (2005) *J. Am. Chem. Soc.*, **127**, 10,848–10,849.

Rowland, S.W. (1979) In *Image Reconstruction from Projections*, Herman, G.T. (Ed.), Springer-Verlag, Berlin.

Schmieder, P., Stern, A.S., Wagner, G. and Hoch, J.C. (1994) *J. Biomol. NMR*, **4**, 483–490.

Shepp, L.A. (1980) *J. Comput. Assist. Tomogr.*, **4**, 94–107.

Swarztrauber, P.N. (1982) In *Parellel Computations*, Rodrigue, G. (Ed.), Academic Press, New York.

Szyperski, T., Wider, G., Bushweller, J.H. and Wuthrich, K. (1993a) *J. Biomol. NMR*, **3**, 127–132.

Szyperski, T., Wider, G., Bushweller, J.H. and Wuthrich, K. (1993b) *J. Am. Chem. Soc.*, **115**, 9307–9308.

Venters, R.A., Coggins, B.E., Kojetin, D., Cavanagh, J. and Zhou, P. (2005) *J. Am. Chem. Soc.*, **127**, 8785–8795.