Digital Perfusion Phantoms for Visual Perfusion Validation

Oleg S. Pianykh[1,2]

OBJECTIVE. Despite the increasingly broad use of perfusion applications, we still have no generally accessible means for their verification: The common sense of perfusion maps and "bona fides" of perfusion software vendors remain the only grounds for acceptance. Thus, perfusion applications are one of a very few clinical tools considerably lacking practical objective hands-on validation.

MATERIALS AND METHODS. To solve this problem, we introduce digital perfusion phantoms (DPPs)—numerically simulated DICOM image sequences specifically designed to have known perfusion maps with simple visual patterns. Processing DPP perfusion sequences with any perfusion algorithm or software of choice and comparing the results with the expected DPP patterns provide a robust and straightforward way to control the quality of perfusion analysis, software, and protocols.

RESULTS. The deviations from the expected DPP maps, observed in each perfusion software, provided clear visualization of processing differences and possible perfusion implementation errors.

CONCLUSION. Perfusion implementation errors, often hidden behind real-data anatomy and noise, become very visible with DPPs. We strongly recommend using DPPs to verify the quality of perfusion applications.

Keywords: deconvolution, digital perfusion phantoms, medical errors, perfusion, perfusion algorithms, perfusion validation tool, quality assurance, truncated singular value decomposition

DOI:10.2214/AJR.11.7061

Received April 13, 2011; accepted after revision January 4, 2012.

[1]Department of Radiology, Beth Israel Deaconess Medical Center, Harvard Medical School, 330 Brookline Ave, Boston, MA 02215. Address correspondence to O. S. Pianykh (opiany@gmail.com).

[2]School of Applied Mathematics and Information Science, National Research University-Higher School of Economics, Moscow, Russia.

AJR 2012; 199:627–634

0361–803X/12/1993–627

Over the past decade, perfusion has become one of the mainstream clinical analysis tools. Blood flow (F_B), blood volume (V_B), mean transit time (T_M), and other perfusion parameters are routinely extracted from temporal imaging sequences. Their values and color maps are used as the basis for critical clinical decisions. Their applications range from emergency departments to pharmaceutical trials. References to perfusion data have become commonplace in diagnostic radiology.

This ever-growing need produced a broad spectrum of perfusion algorithms derived from even broader sets of assumptions and mathematical models [1–12]. Their outcomes depend on parameters defined by their authors and modified by their users to reflect various processing and viewing preferences (e.g., eigenvalue thresholds, input-output functions, degrees of continuity, resolution, color ramps, recirculation points, intensity ranges, and smoothing kernels, to name a few). In addition, all well-known perfusion methods—such as the universally adopted deconvolution approach, discussed later—have intrinsic limitations that lead to yet another layer of assumptions and models to fix them; truncated singular value decomposition (TSVD), Tikhonov regularization, and L curves are examples of deconvolution "fixes" [13]. As a result, we are facing a wide spectrum of processing options affecting every aspect of perfusion analysis.

For that reason alone, perfusion measurements from different software may not correlate very well—even when applied to the same data [9, 14–16]. In fact, they might not correlate at all [17]. Therefore, when a hospital faces the task of upgrading its perfusion tools, it often must choose between two evils: not upgrading (but preserving measurement consistency) or upgrading (but likely losing the consistency with the previous results).

Finally, the problem of diverse methodology only gets worse when one needs to judge the performance of a perfusion algorithm on the basis of real patient data. The varying aspects of real image acquisition—such as noise, dose and contrast levels, scanning protocols, artifacts, pathologies and abnormalities, and many more—largely affect perfusion results, making many comparisons incompatible. As a result, radiologists are frequently puzzled by

strange perfusion outcomes, such as unusual insensitivity to the input data, as shown in Figure 1. With too many factors affecting the final results, a rational analysis becomes virtually impossible.

All this creates a natural need for a robust, nonintrusive, scanner- and patient-free perfusion validation tool that would let one compare the outcomes and accuracy of several perfusion-processing applications as objectively as possible. We introduce digital perfusion phantoms (DPPs) as a solution to this problem. DPP is a simulated DICOM image sequence specifically designed to have known perfusion maps with simple and regular visual patterns. These patterns can be compared with the actual maps computed from a DPP with a perfusion algorithm of choice. The result of this comparison provides an objective way to assess the quality of the chosen perfusion algorithm, software, and protocols.

For the rest of this article, we will assume that we work with perfusion CT; perfusion MRI phantoms can be built in the same way and CT attenuations would usually be replaced with the log of MR intensity [2].

Materials and Methods

Digital Perfusion Phantoms: The Concept

Let S be a temporal DICOM image (I) sequence:

$$S = \{I_0, I_1, \ldots, I_n, \ldots, I_N\},$$

capturing contrast material passing through the volume of interest. The image index n corresponds to the acquisition time t_n, with time t_0 being the unenhanced baseline. Contrast enhancement is reflected in pixel intensity values $I_n(x, y)$, that change with the acquisition time t_n and reproduce contrast density according to the imaging modality (linearly in CT, exponentially in MRI). Perfusion parameters F_B, V_B, and T_M are computed from this observed contrast density at each pixel (x, y) on the basis of the chosen perfusion algorithm and modality. The resulting spatial distributions of parameter values are displayed as perfusion maps, with a preset color ramp; there is no color information in the original perfusion values.

We define *DPP* as a computer-simulated image sequence:

$$S = \{I_1, I_2, \ldots, I_n, \ldots, I_N\},$$

where each temporal pixel value $I_n(x, y) = I_n(x, y; t_n)$ is generated to create predefined visual perfusion map patterns. That is, given F_B, V_B, and T_M values (maps), we build DPP sequence S to produce these maps.

Consider the fundamental perfusion convolution equation, which we refer to as equation 1, that has been adopted by most current perfusion analysis tools:

$$I_n(x, y) = I_n(x, y; t_n) = I_n(x, y) + \int_{\tau=0}^{t_n} R(x, y; \tau)\, K(t_n - \tau)\, d\tau + e(x, y) \quad (1)$$

where $I_0(x, y) = I(x, y; t_0 = 0)$ is the baseline image, $R(x, y; t)$ is the residual function, $K(t)$ is the convolution kernel (found from the arterial input function and venous output function), and $e(x, y)$ is random noise [2, 3, 7, 8]. Then, to build a visual perfusion phantom S, we need to select $I_0(x, y)$, $R(x, y; t)$, and $K(t_n)$ in such a way that $F_B(x, y)$, $V_B(x, y)$, and $T_M(x, y)$ maps would contain predefined regular visual patterns—that is, "digital phantoms." In this respect, DPP is a known solution to equation 1. On the other hand, any perfusion algorithm can be viewed as a numeric solver for equation 1. Hence, the principal concept of DPP: If a solver fails on a known solution, it cannot be trusted in general.

This transforms DPP into a straightforward perfusion validation tool.

Building a Digital Perfusion Phantom

Temporal phantom design—To design DPP sequence S, we used two independent perfusion parameters—F_B and T_M—to then calculate volume according to the central volume principle [18]: $V_B = F_B \times T_M$. Using brain perfusion analogy, we chose to simulate a 45-second perfusion sequence ($n = 45$, $T = t_N = 45$ seconds) with a 1-second interimage delay as follows (Fig. 2A):

$$K(t) = t^a e^{-t/b},$$

where $K(t)$ is the convolution kernel function, t is time, and a and b are positive constants used to define different perfusion flow patterns.

We also chose the following piecewise-linear function:

$$R(t) = F_B \times \max(0, 1 - ct),$$

where $R(t)$ is residual function, F_B is blood flow, c is positive constant, and t is time. We chose these functions because they, first, closely reproduce the shapes of the real $R(t)$ and $K(t)$ extracted from the true perfusion data; and, second, represent popular choices studied in the perfusion literature [2, 11] and therefore are used by perfusion software developers to implement perfusion algorithms.

Positive constants a, b, and c were chosen to match the given T_M map (as defined in the following section) and to produce various temporal patterns over the 45-second time interval (Fig. 2B). Using equation 1, we obtained a closed-form solution for the contrast attenuation $I(x, y; t_n)$ as a function of time. As a result, we defined temporal $I(x, y; t_n)$ corresponding to given F_B, T_M, and V_B.

Spatial phantom design—The main goal of DPP spatial design was to find a pattern that can be easily visualized on the F_B, T_M, and V_B maps. We discovered the following exponential parameterization, which we refer to as equation 2, to achieve that goal:

$$T_M(x, y) = T_0\left(\frac{T_1}{T_0}\right)^{y/d},\ F_B(x, y) = F_0\left(\frac{T_1}{T_0}\right)^{x/d},\ 0 \le x, y \le d \quad (2)$$

where (x, y) are integer pixel coordinates, T_0 and T_1 are maximum and minimum transit times, respectively; F_0 is minimum blood flow; and d is image width or height (we used square images). In this case, the third volume parameter, V_B, has to follow the same exponential equation, which we refer to as equation 3:

$$V_B(x, y) = F_0 T_M = V_0\left(\frac{T_1}{T_0}\right)^{(x+y)/d},\ 0 \le x, y \le d \quad (3)$$

where (x, y) are integer pixel coordinates, V_0 is minimum blood volume, and d is image width or height. Therefore, with respect to the central volume principle, our exponential parameterization (equations 2 and 3) has a unique spatial property: T_M remains constant along the horizontal axis x; F_B, along the vertical axis y; and V_B, on diagonal lines $x + y$ = constant. To make this pattern more apparent, we divided each image axis into 10 equal segments with constant F_B, T_M, and V_B values, thus producing the striped patterns—horizontal in T_M, vertical in F_B, and diagonal in V_B (Fig. 3). Thus, constant-value square tiles can be used to judge perfusion algorithm stability; sharp boundaries between them indicate greater sensitivity to changes, and striped patterns—equation 1 is being correctly implemented.

As a result, we designed a baseline DPP conforming to perfusion convolution and central volume principles and producing regular linear patterns in the perfusion maps. For the remainder of this article, we will be considering only the temporal part of our DPP phantom: the 400 × 400 DPP square area (inside the "skull" outline) changing in time.

Artifacts—In addition to satisfying the most fundamental properties, perfusion algorithms are expected to be numerically stable and precise. Therefore, we decided to complement our baseline DPP with random noise for noise-resistance analysis and thin angular lines for spatial resolution analysis.

One-pixel-thin angular lines were drawn from two phantom corners (Fig. 4B): left-top (point L) and right-top (point R). The lines radiating from L, which we refer to as "L lines," were made with a 25% increase in the underlying phantom intensity and those radiating from R (i.e., "R lines"), with a 50% increase. Many perfusion packages use smoothing to overcome noise, often diluting and losing fine image details. Therefore, thin lines were added to test the spatial resolution of the algorithm.

Random gaussian noise, commonly used in perfusion simulations [19], was also added to test the algorithm's resistance to noise (Fig. 4C). From the many available ways to define noise patterns, we chose noise proportional to the temporal deviation of the contrast enhancement line $I(x, y; t_n)$ in

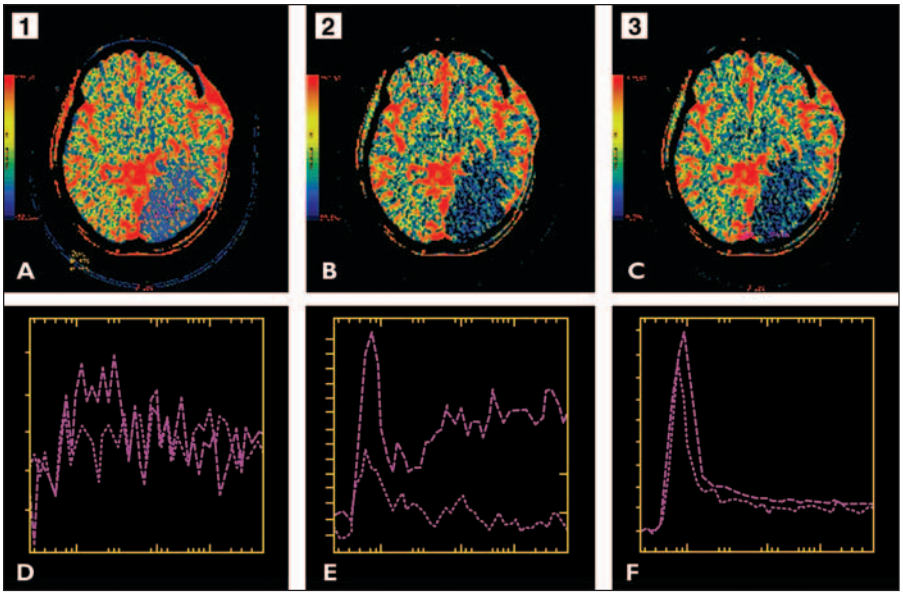

Fig. 1—Perfusion implementation oddities.
A–C, Three perfusion flow maps computed from same brain CT sequence using same settings and same popular perfusion software package. Only difference was in arterial input and venous output point locations; these points were selected correctly for map 3 (**C**) and were completely misplaced on background noise in map 1 (**A**) and on sinus in map 2 (**B**).
D–F, Despite very clear differences in arterial and venous function profiles shown in **D-F** (HU vs. time graphs), flow maps 1, 2, and 3 (**A–C**) look strikingly similar. Even map 1 (**A**), based on random noise input (**D**), looks similar to map 3 (**C**), derived from true flow functions (**F**).

equation 1, thus covering a wide range of noise amplitudes. Noise and thin lines model local intensity changes, with opposite implications: Ideal perfusion software should suppress the noise without losing the fine lines.

As a result, we generated two DPP sequences: baseline DPP_1 (Figs. 3 and 4A) and artifact (lines and noise in Fig. 4C) DPP_2. For all images, we used a 512 × 512 pixel matrix with a large 400 × 400 phantom square in the middle divided into 10 × 10 square tiles (40 × 40 pixels each), with each tile corresponding to a constant triplet of F_B, T_M, and V_B values (Fig. 3). The attenuations (measured in Hounsfield units) in the resulting 400 × 400 phantom matrix ranged from 10 to 510 HU; on the outside, the phantom was surrounded by a simulated skull outline (required in some perfusion packages) and air background (–1000 HU). The interimage time delay was set to 1 second, and all phantom images were stored in DICOM format, conforming to DICOM CT SOP. Thus, the only difference between DPP and real perfusion scans was in the simulated spatial distribution of DPP pixel values (square tiles with constant perfusion values); everything else was consistent with real perfusion data such as CT brain perfusion.

Results

To see DPPs in action, we tested three popular perfusion packages using their brain perfusion option: software A (TeraRecon Perfusion, provided to us via the TeraRecon test server), software B (Perfusion 4, GE Healthcare), and software C (Vitrea 6.1, Vital Images). All three packages claimed perfusion deconvolution implementation [2], thus conforming to equation 1 and our DPP design. Two DPPs were used: baseline DPP_1 (artifact-free) (Fig. 4A) and artifact DPP_2 (with thin lines and noise) (Fig. 4C), as detailed earlier. All three programs received the two DPP sequences as input, with identical arterial input function, venous output function, thresholding, and color ramp settings. The resulting perfusion maps were only adjusted to be in the same color and intensity window for con-

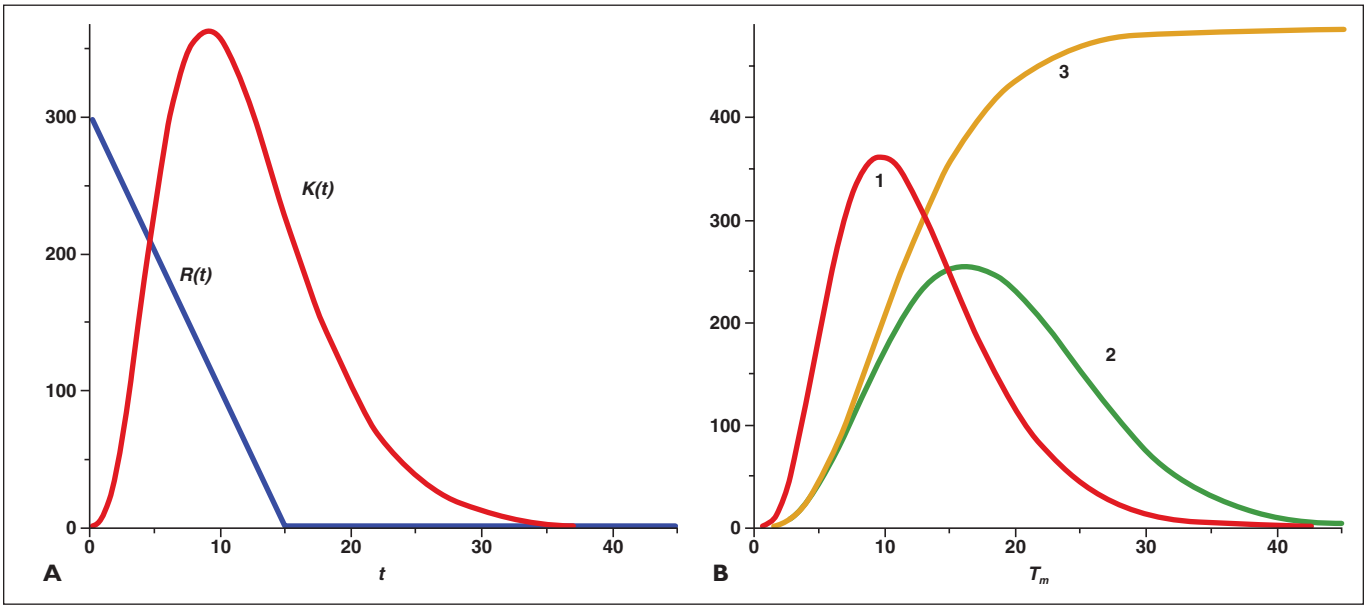

Fig. 2—Perfusion convolution functions.
A, $R(t)$ is the residual function, $K(t)$ is the convolution kernel function, and t is time, used in our digital perfusion phantom. The vertical axis shows the change in HU.
B, Examples of resulting contrast attenuation curves $I(x, y, t_n)$ from equation 1, shown for mean transit time (T_m) parameter set to 1 second (*curve 1*), 20 seconds (*curve 2*), and 45 seconds (*curve 3*).

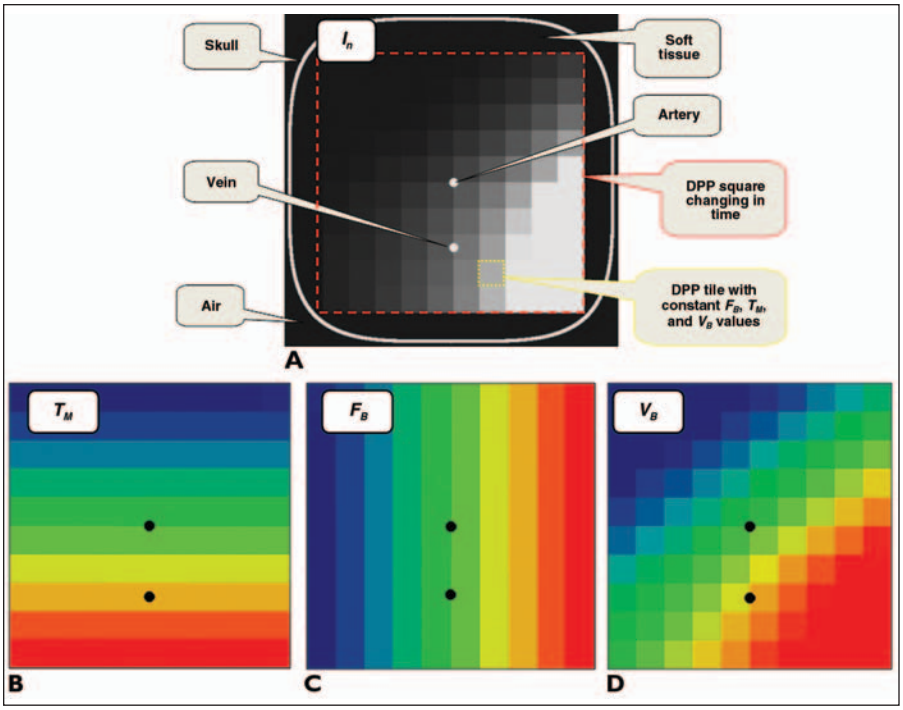

Fig. 3—Digital perfusion phantom (DPP) design.
A, One of phantom temporal images, I_n, with simulated anatomy and 400 × 400 digital perfusion phantom (DPP) square area, changing in time. F_B = blood flow, T_M = mean transit time, V_B = blood volume.
B–D, Perfusion maps for DPP area square constructed with exponential parameterization: mean transit time (T_M, horizontal stripes); blood flow (F_B, vertical stripes), and blood volume (V_B, diagonal stripes).

sistent comparison. The resulting maps are discussed below. Because all perfusion programs offered different (and often incompatible) options, we ran them with the default settings provided by their vendors.

Blood Volume Maps

The baseline DPP_1 was used to test software packages A, B, and C for an expected diagonal pattern in the V_B maps. As one can see from the output DPP_1 maps (Figs. 5A–5C), software B produced the closest match to the expected diagonal pattern, whereas software A and C visibly deviated from the 45° slope and software C deviated from the expected slope linearity (left bottom corner of Fig. 5C corresponding to low-F_B and high-T_M area). In comparison, software B (Fig. 5B) had the lowest dynamic color range: Half of its map was uniformly red, with most intermediate shades essentially lost. The volume map from software A (Fig. 5A) showed much more subtle differentiation in V_B values.

Also note that the square tiles with constant V_B had very visible boundary artifacts in maps from software packages B (Fig. 5B) and C (Fig. 5C), such as the "ghost" line shown with arrow 1 in Figure 5B; however, while tile edges were simply smoothed by software B in Figure 5B, they received overlapping rectangular boundaries by software C (arrow 2 in Fig. 5C). Some perfusion values in the map yielded by software C were clearly misplaced—such as the yellow rectangle surrounded by a red boundary (arrow 3)—raising questions about the correctness of the map and sensitivity of software C to the input.

DPP_2 with thin line and noise artifacts uncovered even more differences between the three perfusion packages (Figs. 5D–5F). Software A preserved a reasonable number of fine lines despite apparent smoothing: One can see the smoothed noise on the square tiles (expected to have constant color). Perfusion smoothing has one principal disadvantage: Smoothed noise starts looking like structural anatomy. Nonetheless, software B clearly used more aggressive smoothing than software A, and removed both noise and fine line details. Artificial colors on tile boundaries (arrow 4 in Fig. 5E) and false spots (arrow 5 in Fig. 5E) presented other unfortunate results of oversmoothing. The thin lines in the top left corner of software B's map (Fig. 5E; low F_B and T_M) mutated into shapeless blots (arrow 6 in Fig. 5E) that could also be confused with anatomic or pathologic findings on real-data maps. Finally, note the rounded corners on the map yielded by software B (arrow 7 in Fig. 5E)—likely another side effect of a large smoothing kernel.

Software C essentially flattened all noise and thin line artifacts; they were completely lost and ignored. Small traces of them can be seen in the light-blue tiles near the diagonal (Fig. 5F). Note that these artifacts also were somewhat "rectagonalized" like the intertile boundaries. In this way, software C turned out to be the most insensitive to the fine image detail given that its map of DPP_2 (Fig. 5F) looks very much like that for DPP_1 (Fig. 5C). Also note the loss of data in the top left corner and bottom right corner, shown by two arrows labeled arrow 8 in Figure 5F, making it look as if software C's map is an overlap of two identical maps shifted along the "green" diagonal.

Ghost lines, blots, smoothed noise, and disfigured edges—so obvious with DPPs—become undistinguishable from true anatomic and pathologic findings in real patient cases. They result in distorted perfusion measurements and can easily lead to incorrect clinical decisions. The dissimilarities in DPP output for all three packages also illustrate that perfusion software from different vendors cannot be expected to deliver the same results [14] and may not even capture the same flow trends.

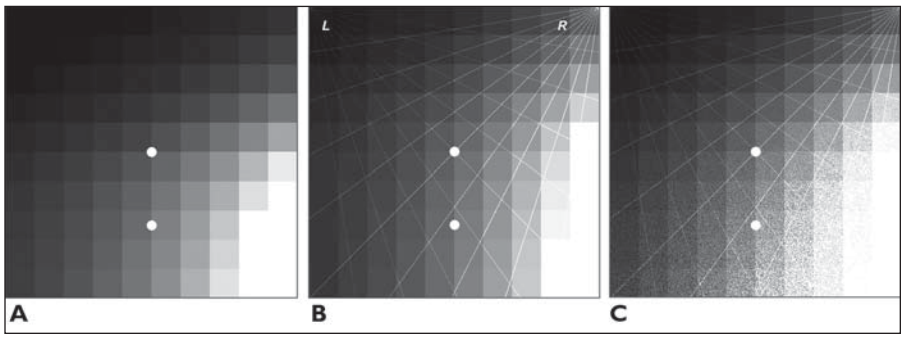

Fig. 4—Simulating artifacts in digital perfusion phantom (DPP) sequence.
A–C, Original DPP (**A**) with addition of thin lines (**B**) and noise artifacts (**C**).

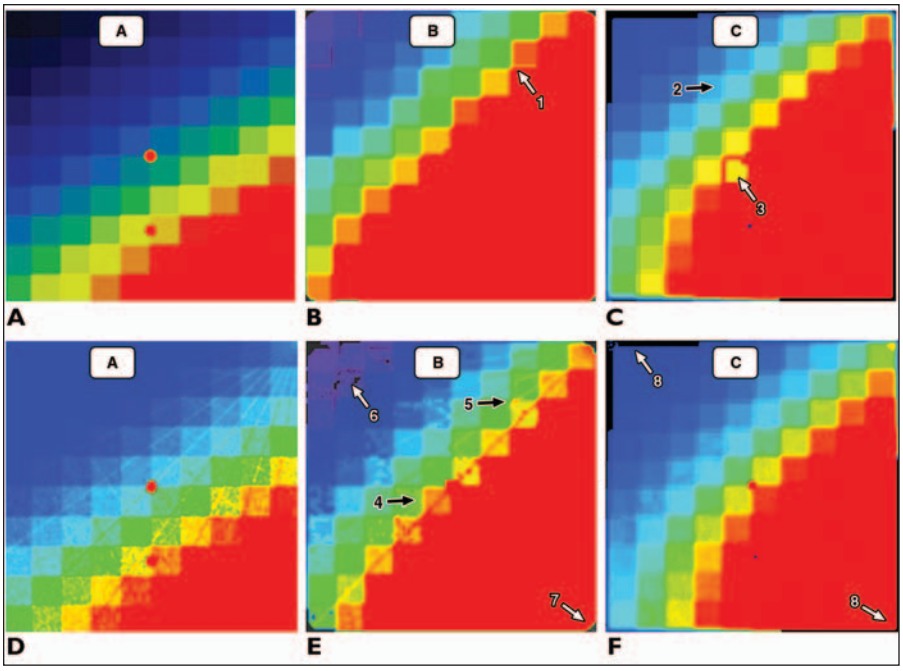

Fig. 5—Digital perfusion phantoms (DPPs) verifying blood volume maps.
A–C, Baseline DPP without artifacts (DPP_1). Blood volume (V_B) maps from perfusion packages A (**A**), B (**B**), and C (**C**) are shown. In **B**, arrow 1 points to "ghost" line. In **C**, arrow 2 shows tile edges received overlapping rectangular boundaries and arrow 3 points to misplaced perfusion value.
D–F, DPP with noise and line artifacts (DPP_2). V_B maps from perfusion packages A (**D**), B (**E**), and C (**F**) are shown. Software B clearly used more aggressive smoothing, and both noise and fine line details were removed. Artificial colors on tile boundaries (*arrow 4,* **E**) and false spots (*arrow 5,* **E**) presented additional unfortunate results of oversmoothing. Thin lines in top left corner of software B's map (low blood flow and mean transit time) mutated into shapeless blots (*arrow 6,* **E**) that can also be confused with anatomic or pathologic findings on maps of real data. Finally, note rounded corners (*arrow 7,* **E**) on software B's map; this feature is likely another side effect of a large smoothing kernel. For software C, note loss of data in top left (*upper arrow 8,* **F**) and bottom right corners (*lower arrow 8,* **F**), making it look as if software C's map is overlap of two identical maps shifted along "green" diagonal.

Blood Flow Maps

The F_B maps shown in Figure 6 were expected to produce vertical stripes (Fig. 3C), but none of the three software packages reached that goal. Software C probably was the closest to the ideal striping, but even it severely deviated from the vertical pattern in the top right corner of Figures 6C and 6F (low-T_M, high-F_B area). For software A and B, nonvertical deviations were apparent for both low-T_M and high-T_M areas. Overall, all three packages provided a perfect illustration of highly uncorrelated perfusion software [14].

Visible distortion of the expected F_B pattern can be explained in the context of deconvolution perfusion analysis. F_B is the hardest to compute with deconvolution: It is often expressed as $R(0)$ or as $\max[R(t)]$. Using either to compute F_B makes F_B very sensitive to noise and algorithm settings, such as eigenvalue truncation threshold. Therefore, the deviations from the vertical stripes in Figure 6 can be somewhat anticipated. This example also makes us aware of the fact that many perfusion deconvolution techniques, such as TSVD, are nothing but the approximations to the ideal solution, often trading perfusion accuracy for higher noise suppression [17]. For that reason, many perfusion deconvolution techniques can significantly diverge from the true solution in some input data ranges. DPPs can clearly identify the areas of this divergence.

DPP_2 with noise and thin line artifacts (Figs. 6D–6F), similar to the V_B case, showed stronger smoothing in software package B (still merging noise in large blots, as shown by arrow 1 in Figure 6E); a large impact of noise on the structural elements in software A (as shown by breaking square tiles in Figure 6D), and nearly complete insensitivity to noise and thin lines in software C (as shown by Figure 6F). Note that in software B, artifacts resulted in further deviation from the vertical pattern, especially in the now-green top right corner of Figure 6E (high-F_B, low-T_M area) where the coloring pattern became essentially diagonal. In software C, we observed the same misplaced yellow tiles (arrow 2 in Fig. 6F)—possible sign of numeric instability of the perfusion algorithm, introducing false perfusion values to the locations where they should not exist.

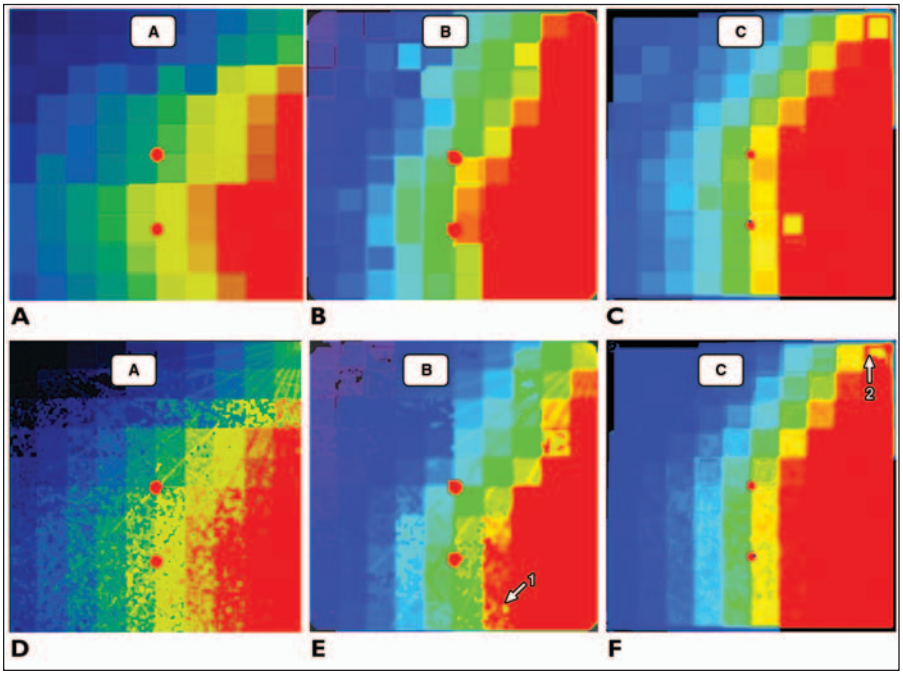

Fig. 6—Digital perfusion phantoms (DPPs) verifying blood flow maps.
A–C, Baseline DPP (DPP_1). Flow volume (F_B) maps from perfusion packages A (**A**), B (**B**), and C (**C**) are shown.
D–F, DPP with artifacts (noise and lines) (DPP_2). F_B maps from perfusion packages A (**D**), B (**E**), and C (**F**) are shown.

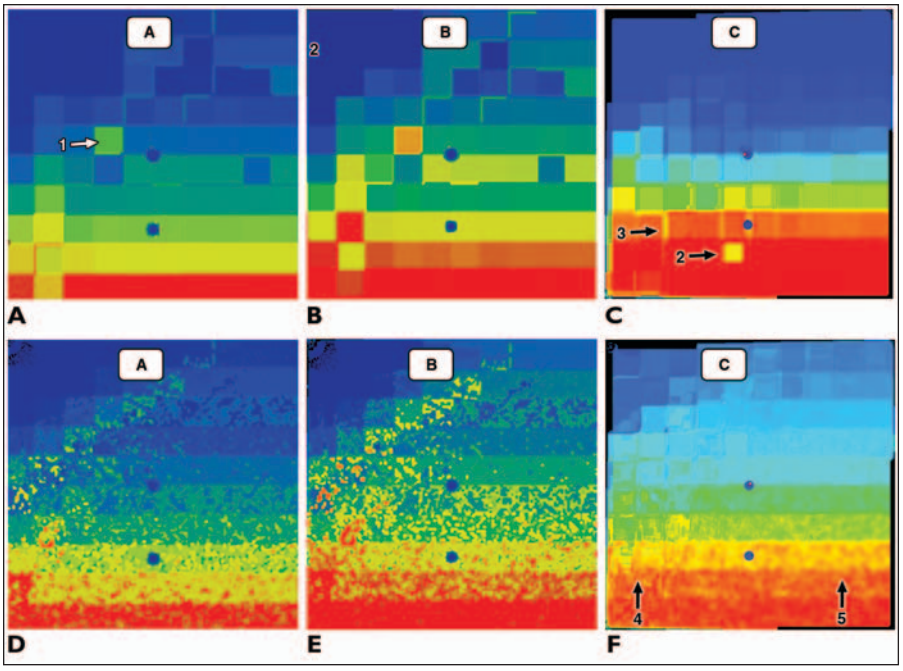

Fig. 7—Digital perfusion phantoms (DPPs) verifying mean transit time maps.
A–C, Baseline DPP (DPP_1). Mean transit time (T_M) maps from perfusion packages A (**A**), B (**B**), and C (**C**) are shown. In **A**, arrow 1 shows misplaced constant-value tile, which resulted from a relatively high T_M value being surrounded by neighbors with visibly lower T_M values. In **C**, arrow 2 shows misplaced T_M value and arrow 3 shows thickened false intertile boundary.
D–F, DPP with artifact (noise and lines) (DPP_2). T_M maps from perfusion packages A (**D**), B (**E**), and C (**F**) are shown.

Mean Transit Time Maps

T_M maps for DPP_1 (Fig. 7) produced rather interesting outcomes for all three implementations. In this case, all three software packages largely deviated from the ideal horizontal stripes, but contrary to the nonlinearities in the F_B maps, deviations in T_M manifested themselves in a more discrete and random fashion—with frequently misplaced constant-value tiles. Consider arrow 1 on the map yielded by software package A (Fig. 7A) as an example of when a relatively high T_M value is surrounded by neighbors with visibly lower T_M values. This behavior is impossible in the ideal DPP solution, where T_M grows monotonically (equation 2), so we have to attribute it to numerically unstable deconvolution implementation. In real clinical cases, perfusion artifacts are usually blamed on noise or poor image acquisition; but Figures 7A–7C, corresponding to the artifact-free DPP_1, clearly show that perfusion imperfections may abound in perfusion algorithms rather than input data.

Note that our DPP design can be used for numeric observations about the perfusion values and their ranges. For instance, consider the misplaced green tile (arrow 1 in Fig. 7A) in the middle of software A's map; this tile is hard to explain rationally given the noise- and artifact-free input data. Exponential parameterization (equation 2), breaking a 45-second time interval into 10 stripes, corresponds to incrementing T_M by a factor of $45^{1/10} = 1.46$, when moving from one stripe to another. The green tile (arrow 1 in Fig. 7A) seems to be shifted three stripes above its line. This three-stripe misplacement corresponds to the factor of $45^{3/10} = 3.13$, which is a substantial 313% error!

It is also instructive to observe that software packages A and B produced nearly identical T_M maps—both in terms of their appearance and processing artifacts—in spite of the clear differences between the A and B maps of V_B and F_B. This finding brings us to another important conclusion: An intervendor perfusion similarity for one perfusion parameter, such as T_M, does not ensure similarity for the other parameters, such as V_B and F_B.

Interestingly enough, adding noise and thin lines (Figs. 7D–7F) contributed to hiding some of the deficiencies observed in the three maps. For example, the misplaced tiles in software A's map were consumed by noise, and the unnaturally looking boundaries between the tiles in software C's map nearly disappeared. Ironically, this result shows another common pitfall with real-life perfusion implementations—that is, when noise and other artifacts "improve" perfusion results by covering the numeric errors in the original perfusion algorithms. Also note that DPP_2 fine-line artifacts were produced

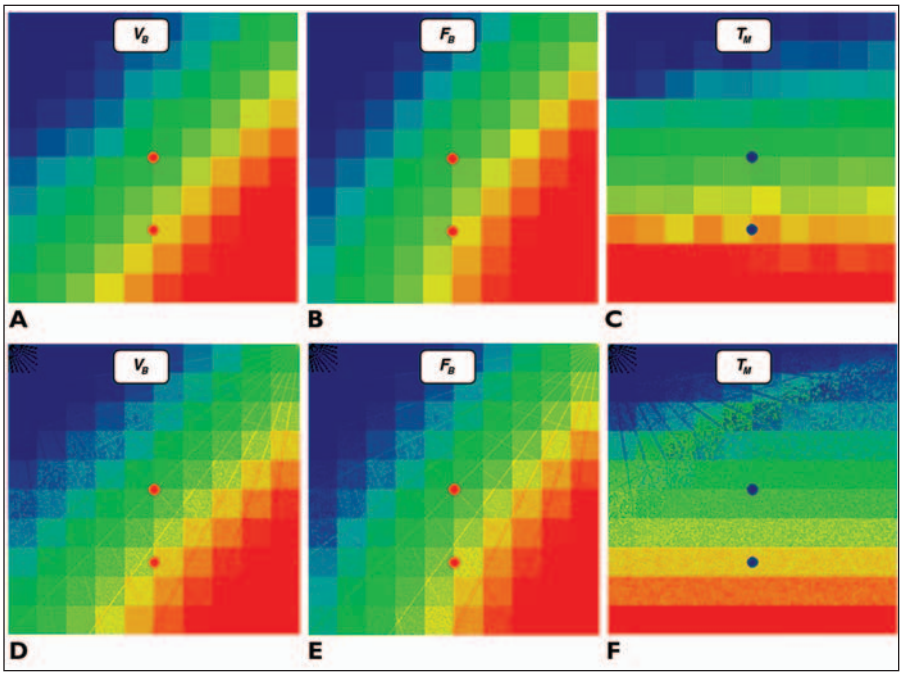

Fig. 8—Perfusion maps for convolution-based digital perfusion phantom (DPP) processed with nonconvolution Axel algorithm. V_B = blood volume, F_B = blood flow, T_M = mean transit time.
A–C, Baseline DPP (DPP_1).
D–F, DPP with artifacts (noise and lines) (DPP_2).

with constant intensity scaling (25% and 50%, respectively), which really should not affect T_M at all. Therefore, the fine lines in the maps yielded by software packages A and B, partially seen in the top left corners of Figures 7D and 7E (lowest F_B and lowest T_M), indicate numeric errors rather than true T_M values.

Finally, the T_M map for software C (Figs. 7C and 7F) exhibits peculiarities of its own. First of all, it looks visibly different from the T_M maps for software A and B and might look like the best approximation to the expected horizontal line pattern. However, it still suffers from misplaced values (arrow 2 in Fig. 7C) and thickened false intertile boundaries (arrow 3 in Fig. 7C). It also shows a very strange response to the same gaussian noise pattern: discrete rectangularlike on the left (arrow 4 in Fig. 7F; low-F_B area), and much smoother and rounded on the right (arrow 5 in Fig. 7F; high-F_B area).

Changes in Implementations, Baseline, and Algorithms

It is natural to ask how DPPs, based on the perfusion convolution model in equation 1, could perform with nonconvolution perfusion algorithms such as Axel's perfusion [1]. We implemented Axel's approach with our software. The results are shown in Figure 8.

As can be seen, the nonconvolution nature of Axel's perfusion analysis changed the observed coloring pattern: The stripes in F_B became diagonal (instead of vertical) with the same diagonal pattern also seen in the upper "blue" area of the T_M map. Nevertheless, the change in global stripping pattern did not (and should not) affect the local continuity of the perfusion value distribution and the regularity of the DPP visual pattern. As a result, we can still use our DPPs in a very consistent way. In fact, all we need to know for any new type of perfusion analysis algorithm is its ideal response to the DPP maps built from equations 2 and 3. Then, the algorithm verification work remains exactly the same as illustrated with software packages A, B, and C. Moreover, DPPs can be used to infer the underlying model (deconvolution vs Axel, for instance). And, with a few numeric measurements on DPP maps, one can even deduce the corresponding model parameters, such as eigenvalue thresholding, thus recovering the entire perfusion implementation. In any event and with any software package we studied, DPPs provide very clear feedback about the algorithm's response to noise, fine artifacts, and different ranges of perfusion values.

Another interesting direction lies in superimposing DPPs with a real baseline anatomy. In particular, this can help us visualize to what extent real shapes and contours affect different perfusion implementations. Figure 9 provides a good illustration for this: We used our DPP_1 phantom with a real skull outline as the baseline image I_0.

Although the overall response to DPP_1 (Figs. 9A–9F) remains the same as expected, note the obvious divergence in software packages B and C close to the baseline shape boundaries. Small noise and edge patterns so visible in software B (arrow 1 in Fig. 9A) completely disappear in software C (Fig. 9D); smooth regions in software B (Fig. 9B) become very noisy and unnaturally rectangular in software C (arrow 2 in Fig. 9E); circular areas with clearly different perfusion values in software C (arrow 3 in Fig. 9D) do not match anything in software B (Fig. 9A). Once again, DPPs become ideal for highlighting all these differences—the differences that would otherwise be blamed on anatomy and noise but that, in reality, correspond to diverging perfusion implementations.

Finally, as a good practical and extreme counterexample, Figure 10 shows DPP maps built with two free perfusion programs found online, which we refer to as software packages D and E. The divergences from the expected DPP patterns are appalling, raising many questions about the correctness and numeric stability of software D and E.

We know that perfusion algorithms may vary, but under no circumstances should they produce random and incoherent maps out of smooth and gradually changing DPP data. Implementation errors, so easy to hide behind real-data anatomy and noise, become very visible with DPPs. We strongly recommend using DPPs as the very first perfusion software test to verify the basic quality of the implementation.

Conclusion

We introduced DPP as an objective visual perfusion validation tool and proved its effectiveness regardless of the underlying perfusion algorithm implementation. Verification of the perfusion processing is the most basic and important step for its acceptance, but DPPs can be used in more complex and quantitative experiments. First, just like any real perfusion data, DPPs correspond to a valid contrast flow,

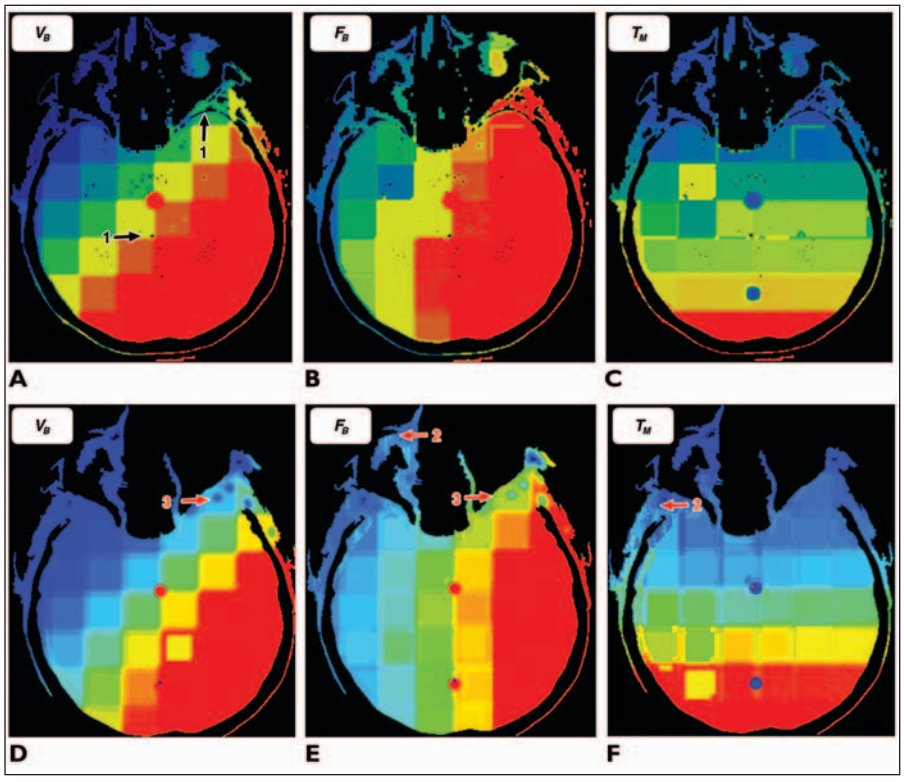

Fig. 9—Baseline digital perfusion phantom (DPP_1), placed over baseline image of skull and processed with two different perfusion packages. Note differences in responses to baseline shape. V_B = blood volume, F_B = blood flow, T_M = mean transit time.
A–C, Software B.
D–F, Software C. Arrows point to different perfusion processing errors in software B and software C packages.

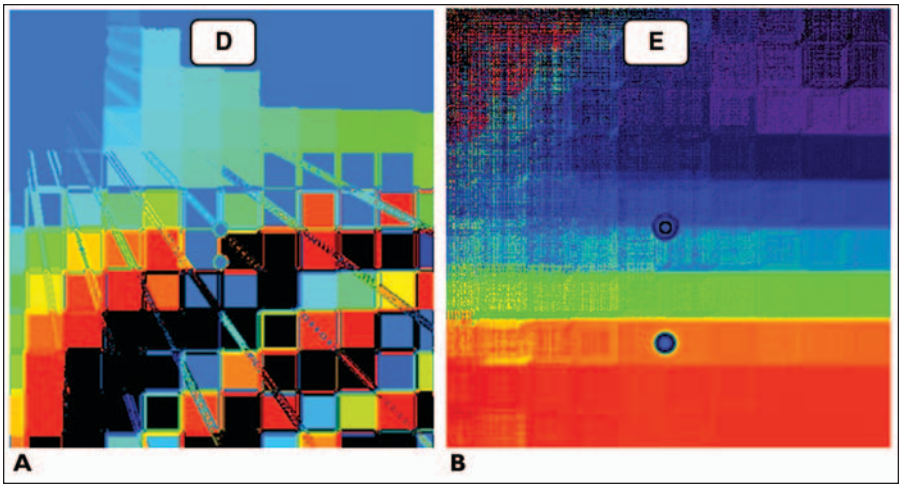

Fig. 10—Extreme cases in digital perfusion phantom (DPP) verification. Note strong and nearly random artifacts on both maps.
A, Blood flow (F_B) map from software package D. Baseline DPP (DPP_1) was used with thin L lines added.
B, Mean transit time (T_M) map from software package E.

producing meaningful values of F_B, T_M, and V_B (Fig. 2). These values can be assessed and compared. However, unlike the real perfusion scans, our DPPs are not affected by scanner settings or choice of patients, pathologic findings, or complex anatomy. This constitutes the principal advantage of DPP, justifying their use for objective perfusion comparison.

Second, our core DPP design does not preclude perfusion researchers and vendors from building their own DPP sequences tuned to reflect more specific perfusion phenomena. However, the DPP_1 sequence we offer provides a universal baseline designed to verify the numeric properties of perfusion analysis tools regardless of perfusion acquisition aspects.

Third, artifacts (i.e., thin lines and noise) in DPP_2 can be altered in strength to find the "breaking point" of given perfusion technique or protocol. Consider the problem of reducing CT radiation exposure: DPPs with increasing noise (corresponding to lower CT exposure) can be used to determine the radiation thresholds for perfusion in low-dose CT.

Fourth, treatment of noise becomes one of the principal factors for judging the quality of perfusion implementation. Strong spatial noise filtering (smoothing) not only removes the important image details, but also often changes the random noise pattern, making it look more like structured anatomy. Strong temporal denoising, such as larger eigenvalue truncation in TSVD, produces visible deviations from the expected perfusion values and their distribution patterns [17]. DPPs can be used to identify these instances, as illustrated.

Fifth, this work should not be viewed as an advertisement of any particular perfusion product and implementation. As follows from our analysis, all software packages have their own advantages and deficiencies. In this case, DPPs provide an objective, vendor-independent perfusion comparison tool.

This interesting discussion is open, and we invite you to participate. The DPP sequence used in this work will be provided on request in the original DICOM format.

References

1. Axel L. Cerebral blood flow determination by rapid-sequence computed tomography: a theoretical analysis. *Radiology* 1980; 137:679–686
2. Østergaard L, Weiskoff RM, Chesler DA, Gyldensten C, Rosen BR. High resolution measurement of cerebral blood flow using intravascular tracer bolus passages. Part I. Mathematical approach and statistical analysis. *Magn Reson Med* 1996; 36:715–725
3. Lee TY. Functional CT: physiological models. *Trends Biotechnol* 2002; 20(suppl):S3–S10
4. Wu Y, An H, Krim H, Lin W. An independent component analysis approach for minimizing effects of recirculation in dynamic susceptibility contrast magnetic resonance imaging. *J Cereb Blood Flow Metab* 2007; 27:632–645
5. Calamante F, Mørup M, Hansen LK. Defining a local arterial input function for perfusion MRI using independent component analysis. *Magn Reson Med* 2004; 52:789–797
6. Patlak CS, Blasberg RG, Fenstermacher JD. Graphical evaluation of blood-to-brain transfer constants from multiple-time uptake data. *J Cereb Blood Flow Metab* 1983; 3:1–7
7. Miles KA, Leggett DA, Bennett GAJ. CT derived Patlak images of the human kidney. *Br J Radiol* 1999; 72:153–158
8. Wirestam R, Andersson L, Østergaard L, et al. Assessment of regional cerebral blood flow by dynamic susceptibility contrast MRI using different deconvolution techniques. *Magn Reson Med* 2000; 43:691–700
9. Calamante F, Gadian DG, Connelly A. Quantification of bolus-tracking MRI: improved characterization of the tissue residue function using Tikhonov regularization. *Magn Reson Med* 2003; 50:1237–1247
10. Koh TS, Wu XY, Cheong LH, Lim CCT. Assessment of perfusion by dynamic contrast-enhanced imaging using a deconvolution approach on regression and singular value decomposition. *IEEE Trans Med Imaging* 2004; 23:1532–1542
11. University of Graz & Technical University of Graz Website. Keeling SL, Bammer R, Kogler T, Stollberger R. On the convolution model of dynamic contrast enhanced magnetic resonance imaging and nonparametric deconvolution approaches: report no. 298. www.kfunigraz.ac.at/imawww/invcon/medimage/kernel.pdf. Published April 2004. Accessed May 7, 2012
12. Rost E, Geske R, Baake M. Signal analysis of impulse response functions in MR and CT measurements of cerebral blood flow. *J Theor Biol* 2006; 240:451–458
13. Hansen PC. Deconvolution and regularization with Toeplitz matrices. *Numer Algorithms* 2002; 29:323–378
14. Kudo K, Sasaki M, Yamada K, et al. Differences in CT perfusion maps generated by different commercial software: quantitative analysis by using identical source data of acute stroke patients. *Radiology* 2010; 254:200–209
15. Konstas AA, Lev MH. CT perfusion imaging of acute stroke: the need for arrival time, delay insensitive, and standardized postprocessing algorithms? *Radiology* 2010; 254:22–25
16. Goh V, Halligan S, Bartram CI. Quantitative tumor perfusion assessment with multidetector CT: are measurements from two commercial software packages interchangeable? *Radiology* 2007; 242:777–782
17. Pianykh OS. Perfusion linearity and its applications in perfusion algorithm analysis. *Comput Med Imaging Graph* 2012; 36:204–214
18. Stewart GN. Researches on the circulation time in organs and on the influences which affect it: parts I–III. *J Physiol* 1893; 15:1–89
19. Knutsson L, Ståhlberg F, Wirestam R. Aspects on the accuracy of cerebral perfusion parameters obtained by dynamic susceptibility contrast MRI: a simulation study. *Magn Reson Imaging* 2004; 22: 789–798