Supporting Information

Theory in different notation

In this section we first describe the general parallel imaging and compressed sensing problems. Subsequently, the Split Bregman algorithm, which is used to solve these problems, is explained. Hereafter, we introduce the preconditioner that is used to speed up the PI-CS algorithm and elaborate on its implementation and complexity.

Parallel Imaging Reconstruction

In parallel imaging with full k-space sampling the data, including noise, is described by the model

$$\mathsf{FS}\mathbf{x} = \mathbf{y}_{\text{full}},$$

where the vector $\mathbf{y}_{\text{full}} \in \mathbb{C}^{N\mathrm{N_c}\times 1}$ contains the fully sampled noisy k-space data sets for $\mathrm{N_c}$ coil channels. Furthermore, $\mathbf{x} \in \mathbb{C}^{N\times 1}$ is the true image (3). Here, $N = m \cdot n$, where m and n define the image matrix size in the x and y-directions, respectively, for a 2D sampling case. Furthermore, $\mathsf{S} \in \mathbb{C}^{N\mathrm{N_c}\times N}$ are stacked diagonal matrices representing complex coil sensitivity maps for each channel. Finally, $\mathsf{F} \in \mathbb{C}^{N\mathrm{N_c}\times N\mathrm{N_c}}$ is a block diagonal matrix where every block is the discrete two-dimensional Fourier transform matrix $\tilde{\mathsf{F}}$ with dimensions $N \times N$. In the case of undersampling, the data is described by the model

$$\mathsf{RFS}\mathbf{x} = \mathbf{y}, \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^{N\mathrm{N_c}\times 1}$ are the undersampled k-space data sets with zeros at non-measured k-space locations. The undersampling pattern is specified by the binary diagonal sampling matrix $\mathsf{R} \in \mathbb{R}^{N\mathrm{N_c}\times N\mathrm{N_c}}$, so that the undersampled Fourier transform is given by RF. Here it is important to note that R reduces the rank of RFS, which means that solving for $\mathbf{x}$ in Eq. (1) is in general an ill-posed problem for each coil and a unique solution does not exist. However, if the individual coil data sets are combined and the undersampling factor does not exceed the number of coil channels, the image $\mathbf{x}$ can in theory be reconstructed by finding the least squares solution, i.e. by minimizing

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\|\mathsf{RFS}\mathbf{x} - \mathbf{y}\right\|_2^2, \tag{2}$$

where $\hat{\mathbf{x}} \in \mathbb{C}^{N\times 1}$ is an estimate of the true image.

Parallel Imaging Reconstruction with Compressed Sensing

In the case of higher undersampling factors, the problem of solving Eq. (2) becomes ill-posed and additional regularization terms need to be introduced to transform the problem into a well-posed problem. Since MR images are known to be sparse in some domains, adding ℓ_1-norm terms is a suitable choice for regularization. The techniques of parallel imaging and compressed sensing are then combined in the following minimization problem

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{\mu}{2} \left\| \mathsf{RFS}\mathbf{x} - \mathbf{y} \right\|_2^2 + \frac{\lambda}{2} \left(\left\| \mathsf{D}_x \mathbf{x} \right\|_1 + \left\| \mathsf{D}_y \mathbf{x} \right\|_1 \right) + \frac{\gamma}{2} \left\| \mathsf{W}\mathbf{x} \right\|_1 \right\}, \tag{3}$$

with μ, λ and γ the regularization parameters for the data fidelity, the total variation, and the wavelet, respectively (8). A total variation regularization constraint is introduced by the first-order derivative matrices D_x, $\mathsf{D}_y \in \mathbb{R}^{N \times N}$, representing the numerical finite difference scheme

$$\begin{aligned} D_x(x)|_{i,j} &= x_{i,j} - x_{i-1,j} \qquad && i = 2,..,m, \;\; j = 1,..,n \\ D_y(x)|_{i,j} &= x_{i,j} - x_{i,j-1} \qquad && i = 1,..,m, \;\; j = 2,..,n \end{aligned}$$

with periodic boundary conditions

$$\begin{aligned} D_x(x)|_{1,j} &= x_{1,j} - x_{m,j} \qquad && j = 1,..,n \\ D_y(x)|_{i,1} &= x_{i,1} - x_{i,n} \qquad && i = 1,..,m \end{aligned}$$

so that D_x and D_y are circulant. A unitary wavelet transform $\mathsf{W} \in \mathbb{R}^{N \times N}$ further promotes sparsity of the image in the wavelet domain.

Split Bregman Iterations

Solving Eq. (3) is not straightforward as the partial derivatives of the ℓ_1-norm terms are not well-defined around 0. Instead, the problem is transformed into one that can be solved easily. In this work, we use Split Bregman to convert Eq. (3) into multiple minimization problems in which the ℓ_1-norm terms have been decoupled from the ℓ_2-norm term, as discussed in detail in (14,24). For convenience, the Split Bregman method is shown in Algorithm 1. The Bregman parameters $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_w$ are introduced by the Bregman scheme and auxiliary variables $\mathbf{d}_x, \mathbf{d}_y, \mathbf{d}_w$ are introduced by writing the constrained problem as an unconstrained problem. The algorithm consists of two loops: an outer loop and an inner loop. In the inner loop (steps 4-11), we first compute the vector $\mathbf{b}$ that serves as a right-hand side for step 5, which is solving an ℓ_2-norm problem. Subsequently, the ℓ_1-norm subproblems are solved using the shrink function in steps 6-8. Hereafter, the residuals for the regularization terms are computed in steps 9-11 and are subsequently fed back into the system by updating the right hand side vector $\mathbf{b}$ in step 5.

Steps 4-11 can be repeated several times, but one or two inner iterations are normally sufficient for convergence. Similarly, the outer loop feeds the residual encountered in the data fidelity term back into the system, after which the inner loop is executed again.

The system of linear equations,

$$\mathsf{A}\hat{\mathbf{x}} = \mathbf{b}, \tag{4}$$

in line 5 of the algorithm follows from a standard least squares problem, where the system matrix is given by

$$\mathsf{A} = \mu\left(\mathsf{RFS}\right)^H \mathsf{RFS} + \lambda\left(\mathsf{D}_x^H\mathsf{D}_x + \mathsf{D}_y^H\mathsf{D}_y\right) + \gamma\mathsf{W}^H\mathsf{W}$$

with right-hand side

$$\mathbf{b} = \mu\left(\mathsf{RFS}\right)^H \mathbf{y}_i + \lambda\left[\mathsf{D}_x^H\left(\mathbf{d}_x^k - \mathbf{b}_x^k\right) + \mathsf{D}_y^H\left(\mathbf{d}_y^k - \mathbf{b}_y^k\right)\right] + \gamma\mathsf{W}^H\left(\mathbf{d}_w^k - \mathbf{b}_w^k\right).$$

In this work we focus on solving Eq. (4), which is computationally the most expensive part of Algorithm 1. It is important to note that the system matrix A remains constant throughout the algorithm and only the right hand side vector $\mathbf{b}$ changes, which allows us to efficiently solve Eq. (4) by using preconditioning techniques.

Algorithm 1 Split Bregman Iteration

1: *Initialize* $\mathbf{y}^{[1]} = \mathbf{y}$, $\mathbf{x}^{[1]} =$ Sum of Squares$(\mathsf{F}^H\mathbf{y})$,
Initialize $\mathbf{b}_x^{[1]}, \mathbf{b}_y^{[1]}, \mathbf{b}_w^{[1]}, \mathbf{d}_x^{[1]}, \mathbf{d}_y^{[1]}, \mathbf{d}_w^{[1]} = \mathbf{0}$
2: **for** $j = 1$ to nOuter **do**
3: **for** $k = 1$ to nInner **do**
4: $\mathbf{b} = \mu\mathsf{S}^H\mathsf{F}^H\mathsf{R}^H\mathbf{y}^{[j]} + \lambda\left[\mathsf{D}_x^H(\mathbf{d}_x^{[k]} - \mathbf{b}_x^{[k]}) + \mathsf{D}_y^H(\mathbf{d}_y^{[k]} - \mathbf{b}_y^{[k]})\right] + \gamma\mathsf{W}^H(\mathbf{d}_w^{[k]} - \mathbf{b}_w^{[k]})$
5: solve $\mathsf{A}\mathbf{x}^{[k+1]} = \mathbf{b}$ with $\mathbf{x}^{[k]}$ as initial guess
6: $\mathbf{d}_x^{[k+1]} = \text{shrink}\left(\mathsf{D}_x\mathbf{x}^{[k+1]} + \mathbf{b}_x^{[k]}, \frac{1}{\lambda}\right)$
7: $\mathbf{d}_y^{[k+1]} = \text{shrink}\left(\mathsf{D}_y\mathbf{x}^{[k+1]} + \mathbf{b}_y^{[k]}, \frac{1}{\lambda}\right)$
8: $\mathbf{d}_w^{[k+1]} = \text{shrink}\left(\mathsf{W}\mathbf{x}^{[k+1]} + \mathbf{b}_w^{[k]}, \frac{1}{\gamma}\right)$
9: $\mathbf{b}_x^{[k+1]} = \mathbf{b}_x^{[k]} + \mathsf{D}_x\mathbf{x}^{[k+1]} - \mathbf{d}_x^{[k+1]}$
10: $\mathbf{b}_y^{[k+1]} = \mathbf{b}_y^{[k]} + \mathsf{D}_y\mathbf{x}^{[k+1]} - \mathbf{d}_y^{[k+1]}$
11: $\mathbf{b}_w^{[k+1]} = \mathbf{b}_w^{[k]} + \mathsf{W}\mathbf{x}^{[k+1]} - \mathbf{d}_w^{[k+1]}$
12: **end for**
13: $\mathbf{y}^{[j+1]} = \mathbf{y}^{[j]} + \mathbf{y}^{[1]} - \mathsf{RFS}\mathbf{x}^{[k+1]}$
14: **end for**

Structure of the System Matrix A

The orthogonal wavelet transform is unitary, so that $\mathsf{W}^H\mathsf{W} = \mathsf{I}$. Furthermore, the derivative operators are constructed such that the matrices $\mathsf{D}_x, \mathsf{D}_y, \mathsf{D}_x^H$ and D_y^H are block circulant with circulant blocks (BCCB). The product and sum of two BCCB matrices is again BCCB, showing that $\mathsf{D}_x^H\mathsf{D}_x + \mathsf{D}_y^H\mathsf{D}_y$ is also BCCB. These type of matrices are diagonalized by the two-dimensional Fourier transformation $\tilde{\mathsf{F}}$, i.e.

$$\mathsf{D}_1 = \tilde{\mathsf{F}}\mathsf{C}\tilde{\mathsf{F}}^H \quad \text{or} \quad \mathsf{D}_2 = \tilde{\mathsf{F}}^H\mathsf{C}\tilde{\mathsf{F}}$$

where C is a BCCB matrix and D_1 and D_2 are diagonal matrices. This motivates us to write the system matrix A in Eq. (4) in the form

$$\begin{aligned}\mathsf{A} &= \tilde{\mathsf{F}}^H\tilde{\mathsf{F}}\mathsf{A}\tilde{\mathsf{F}}^H\tilde{\mathsf{F}} \\ &= \tilde{\mathsf{F}}^H\mathsf{K}\tilde{\mathsf{F}}\end{aligned} \tag{5}$$

with $\mathsf{K} \in \mathbb{C}^{N\times N}$ given by

$$\mathsf{K} = \mu \underbrace{\tilde{\mathsf{F}}\mathsf{S}^H\mathsf{F}^H\mathsf{R}^H\mathsf{R}\mathsf{F}\mathsf{S}\tilde{\mathsf{F}}^H}_{\mathsf{K}_c} + \lambda \underbrace{\tilde{\mathsf{F}}\left(\mathsf{D}_x^H\mathsf{D}_x + \mathsf{D}_y^H\mathsf{D}_y\right)\tilde{\mathsf{F}}^H}_{\mathsf{K}_d} + \gamma \underbrace{\mathsf{I}}_{\mathsf{K}_w}. \tag{6}$$

The term $\mathsf{D}_x^H\mathsf{D}_x + \mathsf{D}_y^H\mathsf{D}_y$ is BCCB, so that K_d in K becomes diagonal. If there is no sensitivity encoding, that is $\mathsf{S} = \mathsf{I}$, the entire K matrix becomes diagonal in which case the solution $\hat{\mathbf{x}}$ can be efficiently found by computing

$$\hat{\mathbf{x}} = \mathsf{A}^{-1}\mathbf{b} = \tilde{\mathsf{F}}^H\mathsf{K}^{-1}\tilde{\mathsf{F}}\mathbf{b} \tag{7}$$

for invertible K. In practice, Fast Fourier Transforms (FFTs) are used for this step. With sensitivity encoding, $\mathsf{S} \neq \mathsf{I}$ and $\mathsf{S}^H\mathsf{F}^H\mathsf{R}^H\mathsf{R}\mathsf{F}\mathsf{S}$ is not BCCB for any i, hence matrix K is not diagonal. In that case we prefer to solve Eq. (4) iteratively, since finding K^{-1} is now computationally too expensive. It can be observed that the system matrix A is Hermitian and positive definite, which motivates the choice for the conjugate gradient (CG) method as an iterative solver.

Preconditioning

A preconditioner $\mathsf{M} \in \mathbb{C}^{N\times N}$ can be used to reduce the number of iterations required for CG convergence (41). It should satisfy the conditions

1. $\mathsf{M}^{-1}\mathsf{A} \approx \mathsf{I}$ to cluster the eigenvalues of the matrix pair around 1, and
2. determination of M^{-1} and its evaluation on a vector should be computationally cheap.

Ideally, we would like to use a diagonal matrix as the preconditioner as this is computationally inexpensive. For this reason, the Jacobi preconditioner is used in many applications with the diagonal elements from matrix A as the input. However, for the current application of PI and CS the Jacobi preconditioner is not efficient since it does not provide an accurate approximate inverse of the system matrix A. In this work, we use a different approach and approximate the diagonal from K in Eq. (6) instead. The motivation behind this approach is that the Fourier matrices in matrix K center a large part of the information contained in $\mathsf{S}^H\mathsf{F}^H\mathsf{R}^H\mathsf{RFS}$ around the main diagonal of K, so that neglecting off-diagonal elements of K has less effect than neglecting off-diagonal elements of A.

For the preconditioner used in this work we approximate A^{-1} by

$$\mathsf{M}^{-1} = \tilde{\mathsf{F}}^H \text{diag}\{\mathbf{k}\}^{-1} \tilde{\mathsf{F}}, \tag{8}$$

where diag{} places the elements of its argument on the diagonal of a matrix. Furthermore, vector $\mathbf{k}$ is the diagonal of matrix K and can be written as

$$\mathbf{k} = \mu\mathbf{k}_c + \lambda\mathbf{k}_d + \gamma\mathbf{k}_w, \tag{9}$$

where $\mathbf{k}_c$, $\mathbf{k}_d$ and $\mathbf{k}_w$ are the diagonals of K_c, K_d and K_w, respectively. Note that K_d and K_w are diagonal matrices already, so that only $\mathbf{k}_c$ will result in an approximation of the inverse for the final system matrix A.

Efficient Implementation of the Preconditioner

The diagonal elements $\mathbf{k}_c$ of $\mathsf{K}_c = \tilde{\mathsf{F}}\mathsf{S}^H\mathsf{F}^H\mathsf{R}^H\mathsf{RFS}\tilde{\mathsf{F}}^H$ can be found by expressing the matrix K_c in terms of the coil elements as

$$\mathsf{K}_c = \underbrace{\tilde{\mathsf{F}}\mathsf{S}_1^H\tilde{\mathsf{F}}^H}_{\mathsf{C}_1^H}\underbrace{\mathsf{R}_1^H\mathsf{R}_1}_{\mathsf{R}_1}\underbrace{\tilde{\mathsf{F}}\mathsf{S}\tilde{\mathsf{F}}^H}_{\mathsf{C}_1} + \ldots + \underbrace{\tilde{\mathsf{F}}\mathsf{S}_{\mathrm{N_c}}^H\tilde{\mathsf{F}}^H}_{\mathsf{C}_{\mathrm{N_c}}^H}\underbrace{\mathsf{R}_{\mathrm{N_c}}^H\mathsf{R}_{\mathrm{N_c}}}_{\mathsf{R}_{\mathrm{N_c}}}\underbrace{\tilde{\mathsf{F}}\mathsf{S}\tilde{\mathsf{F}}^H}_{\mathsf{C}_{\mathrm{N_c}}}, \tag{10}$$

where S_i and R_i are the sensitivity map and sampling pattern for coil i, respectively. Now, the diagonal elements $\mathbf{k}_{c;i}$ of $\mathsf{K}_{c;i} = \underbrace{\tilde{\mathsf{F}}\mathsf{S}_i^H\tilde{\mathsf{F}}^H}_{\mathsf{C}_i^H}\underbrace{\mathsf{R}_i^H\mathsf{R}_i}_{\mathsf{R}_i}\underbrace{\tilde{\mathsf{F}}\mathsf{S}\tilde{\mathsf{F}}^H}_{\mathsf{C}_i}$ for a certain i are found by noting that $\mathsf{C}_i = \tilde{\mathsf{F}}\mathsf{S}_i\tilde{\mathsf{F}}^H$ is in fact a BCCB matrix. Hence, the diagonal elements $\mathbf{k}_{c;i}$ can now be found on the diagonal of $\mathsf{C}_i^H\mathsf{R}_i\mathsf{C}_i$, so that

$$\mathbf{k}_{c;i} = \sum_{j=1}^{N} \mathbf{e}_j \left(\mathbf{c}_{j;i}^H \mathsf{R}_i \mathbf{c}_{j;i}\right),$$

with $\mathbf{c}_{j;i}^H$ being the j^{th} row of matrix C_i^H and $\mathbf{e}_j$ the j^{th} standard basis vector. Note that the scalar $\left(\mathbf{c}_{j;i}^H \mathsf{R}_i \mathbf{c}_{j;i}\right)$ is the j^{th} entry of vector $\mathbf{k}_{c;i}$. Since R_i is a diagonal matrix which is equal for each coil element, we can write it as $\mathsf{R}_i = \text{diag}\{\mathbf{r}\}$ and therefore

$$\begin{aligned}
\mathbf{k}_{c;i} &= \sum_{j=1}^{N} \mathbf{e}_j \left(\mathbf{c}_{j;i}^H \circ \mathbf{c}_{j;i}^T\right) \mathbf{r} \\
&= \begin{bmatrix} \mathbf{c}_{1;i}^H \circ \mathbf{c}_{1;i}^T \\ \mathbf{c}_{2;i}^H \circ \mathbf{c}_{2;i}^T \\ \vdots \\ \mathbf{c}_{N;i}^H \circ \mathbf{c}_{N;i}^T \end{bmatrix} \mathbf{r} \\
&= \left(\mathsf{C}_i^H \circ \mathsf{C}_i^T\right) \mathbf{r}, \qquad (11)
\end{aligned}$$

where $\circ$ denotes the element-wise (Hadamard) product. Since the element-wise product of two BCCB matrices is again a BCCB matrix, the circular convolution theorem tells us (42,43) that

$$\tilde{\mathsf{F}}\mathbf{k}_{c;i} = \tilde{\mathsf{F}}\left[\left(\mathbf{c}_{1;i}^H \circ \mathbf{c}_{1;i}^T\right)^T * \mathbf{r}\right] = \tilde{\mathsf{F}}\left[\left(\mathbf{c}_{1;i}^H \circ \mathbf{c}_{1;i}^T\right)^T\right] \circ \tilde{\mathsf{F}}\mathbf{r}.$$

The resulting matrix vector product in Eq. (11) can now be efficiently computed as

$$\mathbf{k}_{c;i} = \tilde{\mathsf{F}}^H \left\{\left[\tilde{\mathsf{F}}\left(\mathbf{c}_{1;i}^H \circ \mathbf{c}_{1;i}^T\right)^T\right] \circ \tilde{\mathsf{F}}\mathbf{r}\right\}. \qquad (12)$$

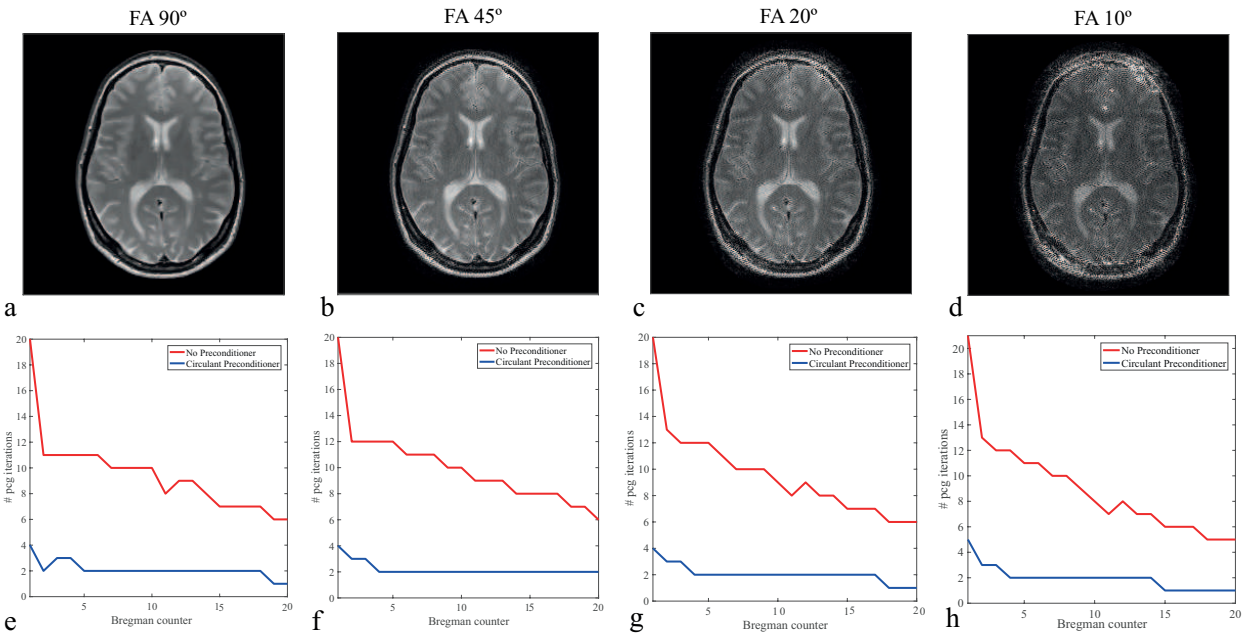
Finally, the diagonal elements $\mathbf{d}$ of the diagonal matrix D with structure $\mathsf{D} = \tilde{\mathsf{F}}\mathsf{C}\tilde{\mathsf{F}}^H$ can be computed efficiently by using $\mathbf{d} = \tilde{\mathsf{F}}\mathbf{c}_1$, where $\mathbf{c}_1$ is the first row of C. Therefore, the first row $\mathbf{c}_{1;i}^H$ of matrix C_i^H is found as $\left(\mathbf{c}_{1;i}^H\right)^T = \tilde{\mathsf{F}}^H \left(\mathbf{s}_i^H\right)^T$, with $\mathbf{s}_i^H$ a row vector containing the diagonal elements of matrix S_i. For multiple coils Eq. (12) becomes

$$\mathbf{k}_c = \tilde{\mathsf{F}}^H \left\{\left[\tilde{\mathsf{F}} \sum_{i=1}^{\mathrm{N_c}} \left(\mathbf{c}_{1;i}^H \circ \mathbf{c}_{1;i}^T\right)^T\right] \circ \tilde{\mathsf{F}}\mathbf{r}\right\}, \qquad (13)$$

where the action of the Fourier matrix on a vector can be efficiently computed using the FFT. Since $\mathsf{D}_x^H \mathsf{D}_x + \mathsf{D}_y^H \mathsf{D}_y$ is BCCB, the elements of $\mathbf{k}_d$ can be quickly found by evaluating $\mathbf{k}_d = \tilde{\mathsf{F}}\mathbf{t}_1$,

where $\mathbf{t}_1$ is the first row of $\mathsf{D}_x^H \mathsf{D}_x + \mathsf{D}_y^H \mathsf{D}_y$. Finally, the elements of $\mathbf{k}_w$ are all equal to one, since K_ω is the identity matrix.

Supporting Figures



Supporting Figure S1. The performance of the preconditioner for different SNR levels. Prospectively undersampled data sets were obtained for different SNR levels by varying the flip angle from 10 to 90 degrees. The difference in the number of CG iterations needed until convergence with and without preconditioner for different SNR levels is negligible.