An Induced Dimension Reduction Algorithm to Approximate Eigenpairs of Large Nonsymmetric Matrices

Reinaldo Astudillo[*,†] and Martin B. van Gijzen[*]

[*]*Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD, Delft, The Netherlands*
[†]*Facultad de Ciencias, Escuela de Computación, Universidad Central de Venezuela, Los Chaguaramos, 1040, Caracas, Venezuela*

Abstract. This work presents an algorithm to approximate eigenpairs of large, sparse and nonsymmetric matrices based on the Induced Dimension Reduction method (IDR(s)) introduced in [1]. We obtain a Hessenberg relation from IDR(s) computations and in conjunction with Implicitly Restarting and shift-and-invert techniques [2] we created a short recurrence algorithm to approximate eigenvalues and its corresponding eigenvectors in a region of interest.

Keywords: Krylov space method, induced dimension reduction, implicitly restarting, large nonsymmetric eigenvalue problem.
PACS: 02.60.-x

INTRODUCTION

The Induced Dimension Reduction method (IDR(s)) [1] is a Krylov subspace method to approximate the solution of linear systems of equations,

$$Ax = b,$$

in particular when the coefficient matrix $A \in \mathbb{R}^{n\times n}$ is large, nonsymmetric and sparse. This method has obtained attention and different variants have been proposed to improved its convergence and numerical stability, for example [3, 4, 5]. Recently, in [6], the IDR(s) method has been adapted to approximate eigenpairs (λ, x) of the matrix A, i.e.

$$Ax = \lambda x, \qquad \text{with} \qquad \lambda \in \mathbb{C}, \text{ and } x \neq 0 \in \mathbb{C}^n. \tag{1}$$

This contribution continues the line of research of approximating eigenvalues and eigenvectors. In next section, we propose a method to obtain an underlying Hessenberg relation from the IDR(s) calculations, then we explain how to combine this method with Implicitly Restarting, and shift-and-invert strategies, in order to approximate specific portions of interest of the spectrum of the matrix A. The last two sections present a numerical example to illustrate the performance of our proposed algorithm and the conclusions.

IDR PROCESS AND IDR FACTORIZATION

In this section we present briefly the ideas behind IDR(s), and describe how to obtain an underlying Hessenberg relation can be used to approximate the spectral information of a matrix. In the first place, let us consider the following theorem ([7, 1]):

Theorem 1. *Let $P = [p_1, p_2, p_3, \ldots, p_s]$ be an $n \times s$ matrix, I the identity matrix of size n, and let $\{\mu_j\}$ be a sequence in $\mathbb{C}$. With $\mathcal{G}_0 \equiv \mathbb{C}^n$, define*

$$\mathcal{G}_{j+1} \equiv (A - \mu_{j+1} I)(\mathcal{G}_j \cap P^{\perp}) \quad j = 0, 1, 2 \ldots$$

If $P^{\perp}$ does not contain an eigenvector of A, then, for all $j = 0, 1, 2 \ldots$ we have that

1. *$dim\mathcal{G}_{j+1} < dim\mathcal{G}_j$ unless $\mathcal{G}_j = \{0\}$.*

Proof. See [1, 7]. □

11th International Conference of Numerical Analysis and Applied Mathematics 2013
AIP Conf. Proc. 1558, 2277-2280 (2013); doi: 10.1063/1.4825994

The basic idea of IDR(s)) to solve linear systems is to force the residual vector $r_k = b - Ax_k$ to be in the sequence of nested subspaces $\mathscr{G}_j$, while in parallel extract the approximate solution vector x_k. In [1] the authors describe the following steps to create a vector in the subspace $\mathscr{G}_{j+1}$.

- First of all, assume that the $s+1$ vectors $w_{i-s}, w_{i-s-1}, \ldots, w_i$ belong to $\mathscr{G}_j$. Consequently any linear combination of these vectors is in $\mathscr{G}_j$, in particular $v = w_i - \sum_{l=1}^{s} c_l w_{i-l}$.
- In order to determine the values c_l, impose the condition that $v \in P^{\perp}$, and this leads to the linear system of order s:
$$(P^T[w_{i-s}, w_{i-s+1}, \ldots, w_{i-1}])c = P^T w_i$$
- Finally the vector in $\mathscr{G}_{j+1}$ is:
$$w_{i+1} = (A - \mu_{j+1} I)\left(w_i - \sum_{l=1}^{s} c_l w_{i-l}\right) \tag{2}$$

From the Eq. (2), we have:

$$Aw_i = w_{i+1} + \mu_{j+1} w_i - \mu_{j+1} \sum_{l=1}^{s} c_l w_{i-l} + \sum_{l=1}^{s} c_l A w_{i-l}$$

Setting $W(:, j:k) = [w_j, w_{j+1}, \ldots, w_k]$, and assuming that Aw_{i-l} can be written as a linear combination of the vectors $w_1, w_2, w_{i-l}, w_{i-l+1}$, for $i = 1, 2, \ldots, i-1$, this is: $Aw_{i-l} = W(:, 1:i-l+1)H(:, i-l)$, we have that

$$Aw_i = W(:, 1:i+1)\left(\begin{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \\ -\mu_{j+1}\begin{bmatrix} c \end{bmatrix} \\ \mu_{j+1} \\ 1 \end{bmatrix} + \sum_{l=1}^{s} c_l H(:, i-l)\right) = W(:, 1:i+1)H(:, i+1) \tag{3}$$

Then to create a new vector in $\mathscr{G}_{j+1}$, we need $s+1$ vectors in the previous subspace $\mathscr{G}_j$. The **IDR(*s*,*m*) process** is outlined in the Algorithm 1, this procedure creates $s+1$ vectors in every subspace $\mathscr{G}_j$ for $j = 1, 2, \ldots, m-1$.

Algorithm 1 IDR(s,m) Process applied to a matrix A with orthogonalization

Given $s, m \in \mathbb{N}$, $P \in \mathbb{R}^{n \times s}$, $W \in \mathbb{C}^{n \times s+1}$ and $H \in \mathbb{C}^{s+1 \times s}$, such that $AW(:, 1:(s+1)-1) = W(:, 1:(s+1))\bar{H}_{(s+1)}$
$i = s+1$ ▷ pointer to the newest vector
for $j = 1, \ldots, m-1$ **do**
  Choose μ_j
  for $k = 0, \ldots, s$ **do** ▷ Create $s+1$ vector in $\mathscr{G}_{j+1}$
    Solve the $s \times s$ linear systems

$$(P^T[w_{i-s}, w_{i-s+1}, \ldots, w_{i-1}])c = P^T w_i$$

    $v = w_i - \sum_{l=1}^{s} c_i w_{i-l}$ ▷ $v \in \mathscr{G}_j \cap P^{\perp}$
    $w_{i+1} = (A - \mu_j)v$ ▷ New vector in $\mathscr{G}_{j+1}$
    $W = [w_1, w_2, \ldots, w_i, w_{i+1}]$ ▷ Update the IDR factorization
    Create the $i+1$-th column of the matrix H according to (3).
      Define $\beta_{i-l} = \langle w_i, w_{i-l} \rangle$ for $l = 1, 2, \ldots, k$, $\beta_i = \|w_{i+1} - \sum_{l=1}^{k} \beta_{i-l} w_{i-l}\|$.
      $w_{i+1} = (w_{i+1} - \sum_{l=1}^{k} \beta_{i-l} w_{i-l})/\beta_i$
      Update $H(i-l, i) = H(i-l, i) + \beta_{i-l}$ for $l = 1, 2, \ldots, k$ and $H(i+1, i) = \beta_i$
    $i = i+1$
  end for
end for

Based on the idea proposed in the section 4.3 of [1] and implemented with encouraging results in [4], we orthogonalize the vectors in each $\mathscr{G}_j$, to this we added the lines 11 to 14 of the Algorithm 1. This IDR(s,m) process creates a Hessenberg relation that we call the **IDR factorization**:

$$AW(:,1:m(s+1)-1) = W(:,1:m(s+1))\bar{H}_{m(s+1)} \tag{4}$$

where $W(:,1:m(s+1)) \in \mathbb{C}^{n\times m(s+1)}$ is a matrix which has m orthogonal blocks of size $s+1$, and a rectangular upper-Hessenberg matrix $\bar{H} \in \mathbb{C}^{m(s+1)\times m(s+1)-1}$. The eigenvalues of the square upper-Hessenberg matrix $H_{m(s+1)}$, obtained from deleting the last row of $\bar{H}_{m(s+1)}$, are divided into two sets: $m-1$ eigenvalues are exactly the parameters μ_j for $j = 1,\ldots,m-1$, and the other $m\times s+1$ eigenvalues are the Ritz values that approximate the eigenvalues of the matrix A (see [6] for details). The IDR factorization is a standard Hessenberg relation which can be used to approximate eigenvalues and eigenvectors of a sparse matrix. Also, this standard Hessenberg relation is suitable for the implementation of the Implicitly Restarted scheme to be discussed in next section.

RESTARTING THE IDR FACTORIZATION

Algorithm 1 has as input parameters an initial Hessenberg relation of size s. This initialization can used with the purpose of restarting the IDR process. Algorithm 2, explains how to combine the IDR process with the Implicitly Restarting technique proposed by D. C. Sorensen in [2].

Algorithm 2 Implicitly restarting of an IDR factorization

1: Given an initial Hessenberg relation of size s.
2: Expand the initial factorization using Algorithm 1, to obtain the IDR factorization of m subspaces:

$$AW(:,1:m(s+1)-1) = W(:,1:m(s+1))\bar{H}_{m(s+1)}$$

3: Reorder the IDR factorization, such that the wanted eigenvalues are contained in the submatrix $H(1:s,1:s)$.
4: Truncate the IDR factorization to obtain the new Hessenberg relation of size s.
5: Test convergence. If no convergence **go to 2** with the new Hessenberg relation else **return**

The reordering technique is accomplished using the QR iteration with exact shifts (see [2, 8] for more details). In several applications it is important to find eigenvalues and its corresponding eigenvectors in a specific region of the complex plane, for example, the eigenvalues with largest real part for stability analysis, or the nearest eigenvalues to a given point for vibrational analysis. To achieve this, one can combine the Implicitly restarting with a shift-and-invert strategy, this is, apply Algorithm 2 to the matrix $(A-\sigma I)^{-1}$ instead of A to approximate the eigenvalues around σ.

A NUMERICAL EXAMPLE

We present a numerical example to illustrate the performance of our proposed algorithm. This experiment was carried out using Matlab 8.0 (R2012b) on a computer with I7 Intel processor 2.4Ghz, 4GB of memory running under GNU/Debian Linux. We compare IDR(s, m) with the Implicitly Restarting Arnoldi method (IRAM(k,p)) proposed in [2]. Table 1 describes the input parameters. Taking into account scaling effect of the matrix $W(:,1:m(s+1))$ over the residual (see discussion in [4]), we propose as stopping criteria for the IDR algorithm:

$$\sqrt{m}\|w_{m(s+1)+1}\|\sqrt{\sum_{i=1}^{s}(y^i_{m(s+1)})^2} < \varepsilon,$$

where $\varepsilon = 10^{-9}$, $y^i_{m(s+1)}$ is the last component of the eigenvector y^i associated with the eigenvalue θ_i of the matrix $H_{m(s+1)}$. In the case of IRAM we ask for the same tolerance ε (see [8] for details).

The matrix arises from the finite difference discretization of the 2D Schrödinger equation. This equation models the energy levels of the confined hydrogen atom, and is given by

$$-u''(x,y) - \frac{2u(x,y)}{\|(x,y)\|} = \lambda u(x,y) \qquad (x,y) \in (-16,16)\times(-16,16) \tag{5}$$

TABLE 1. Description of the parameters used by IRAM and IDR.

IRAM(k,p)	$k \in \mathbb{N}$ number of wanted eigenvalues. $p \in \mathbb{N}$ max dimension of the search subspace, recommended $2k+2$ (see [8]). the initial vector was chosen randomly.
IDR(s,m)	$s \in \mathbb{N}$ has to be a number greater of equal to the number of wanted eigenvalues. $m \in \mathbb{N}$ the number or $\mathscr{G}_j$ subspaces to be created. $m(s+1)$ max dimension of the searching subspace. $P \in \mathbb{R}^{n\times s}$ was chosen randomly. $W \in \mathbb{R}^{n\times s+1}$ and $\bar{H} \in \mathbb{R}^{s+1\times s}$ such that $AW(:,1:s) = W\bar{H}$ obtained from Arnoldi [9]. $\mu_j \in \mathbb{C}$ we compute the median of the first s Ritz values from the initial Hessenberg relation.

TABLE 2. Comparison between IRAM and IDR asked for the 16 leftmost eigenvalues

Method	**Restarts**	**Time (sec.)**	**Residual bound**	**Max. difference from the Ritz values of IRAM**
IRAM($k=16$,$p=34$)	10	10.8425	2.69778e-11	*
IDR($s=16$,$m=2$)	10	9.2144	3.58074e-10	3.22872e-06
IDR($s=16$,$m=3$)	6	10.5782	6.22949e-11	8.1244e-09
IDR($s=16$,$m=4$)	4	10.4732	3.11576e-11	7.64288e-10

We use a nonuniform mesh refined near the origin and obtain a matrix of size 44100×44100. We want to approximate the 16 leftmost eigenvalues. We apply IRAM and IDR to the matrix $(A-\sigma I)^{-1}$, where $\sigma = -2.1$. Table 2 shows a comparison of the performance of IRAM(k,p) and IDR(s,m).

CONCLUSIONS

The main contribution of this work is: we presented an IDR based algorithm to approximate eigenpairs of matrices that use the Implicitly Restarting technique and shift-and-invert strategy. IDR in Table 2 shows competitive results with respect IRAM. These results can be explained by the fact that IDR is a short recurrence method. In order to create a new vector in the Krylov subspace, IDR uses at most $2s$ orthogonalizations, while in the Arnoldi based methods, like IRAM, the computational cost increases in every iteration. Therefore using IDR is possible to create larger dimension subspaces with less computational effort. This interesting fact can be exploited, for example in some applications when only the eigenvalues are required. On the other hand, IDR computes a Krylov subspace basis that is only locally orthogonal, this can affect the convergence speed or numerical stability. To achieve similar results with IRAM, IDR might therefore require a subspace of larger dimension, and consequently more memory.

ACKNOWLEDGMENTS

The authors acknowledge Dr. Domenico Giordano for the discussions about Schrödinger equation, and the referees for their valuable comments. This work was financially supported by Space Research and Technology Centre of the European Space Agency (ESA-ESTEC).

REFERENCES

1. P. Sonneveld, and M. B. van Gijzen, *SIAM J. Scientific Computing* **31**, 1035–1062 (2008).
2. D. C. Sorensen, *SIAM J. Matrix Anal. Appl.* **13**, 357–385 (1992).
3. V. Simoncini, and D. B. Szyld, *SIAM J. Scientific Computing* **32**, 1898–1912 (2010).
4. M. B. van Gijzen, G. L. Sleijpen, and J.-P. M. Zemke, Flexible and multi-shift induced dimension reduction algorithms for solving large sparse linear systems, Tech. rep., 2011-06, DIAM, Delft University of Technology (2011).
5. M. B. van Gijzen, and P. Sonneveld, *ACM Transactions on Mathematical Software* **38**, 5:1–5:19 (2011).
6. M. H. Gutknecht, and J.-P. M. Zemke, *SIAM J. Matrix Anal. Appl.* **34**, 283–311 (2013).
7. G. L. Sleijpen, P. Sonneveld, and M. B. van Gijzen, *Applied Numerical Mathematics* **60**, 1100–1114 (2010).
8. R. B. Lehoucq, D. C. Sorensen, and C. Yang, ARPACK Users' Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods. (1997).
9. W. E. Arnoldi, *Quarterly of Applied Mathematics* **9**, 17–29 (1951).