Singular value decomposition

In mathematics, the singular value decomposition (SVD) of a matrix is a fundamental tool for both theoretical and computational uses. To introduce this concept, you may use the help of the following guideline and bibliography:

Guideline

- Define the full/thin SVD, prove its existence.
- Give and prove some basic properties of the SVD (rank, range, null space ...).
- Relate the singular values with eigenvalues, the spectral norm, and the Frobenius norm.
- Give an insight into the algorithm that computes the SVD of a matrix.

References

[1] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

General information can be found on the Seminar website.
Last edited on 30th January 2025.

Low-rank approximation

Low-rank matrix approximations play a central role in todays data analysis and scientific computing. Large matrices M of size $m \times n$ are common in applications since the data often consists of m objects described by n features. In many cases, an important step in the data analysis is to build a compressed representation of M that may be easier to manipulate and analyze. To introduce the concept of low rank approximation of a matrix, you may use the help of the following guideline and bibliography:

Guideline

- Explain why it is interesting, in terms of storage and computational complexity, to have a matrix of low rank.
- State the Eckart-Young-Mirsky theorem and prove it for the spectral and Frobenius norms. Discuss the unicity and the error bound of a best rank-r approximation for both the spectral and Frobenius norm.
- Present the randomized SVD algorithm and state the error bounds.
- Present the randomized subspace iteration algorithm and state the error bound. Explain why the randomized SVD algorithm works well for matrices whose singular values exhibit some decay.

References

[1] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[2] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211218, 1936.

[3] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 56 of *Springer Series in Computational Mathematics*. Springer, Cham, second edition, 2019.

Introduction to tensors

Tensors are mathematical objects widely used in many fields such as physic, chemistry and data science, among others. To introduce this object, you may use the help of the following guideline and bibliography:

Guideline

- Define the Kronecker product for matrices. State and prove the basic properties of the Kronecker product. Show how to derive efficiently the classical decompositions for several matrices being linked by Kronecker (i.e. QR decomposition, SVD, ...).
- Define the Hadamard and Khatri-Rao product. Show their basic properties.
- Define the vectorization. Show how it is linked to the matrix product via the Kronecker product.
- Define the matricization of a tensor.
- Define the n-mode product of a tensor. State the basic properties of the n-mode product. Express the n-mode matricization of a tensor when it is multiplied by several matrices along several modes.
- Define the contraction of two tensors. State the basic properties of the contraction of two tensors and introduce the diagrammatic notation.

References

[1] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[2] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 56 of *Springer Series in Computational Mathematics*. Springer, Cham, second edition, 2019.

[3] Benjamin Huber, Reinhold Schneider, and Sebastian Wolf. A randomized tensor train singular value decomposition. In *Compressed sensing and its applications*, Appl. Numer. Harmon. Anal., pages 261–290. Birkhäuser/Springer, Cham, 2017.

[4] Tamara G. Kolda. Multilinear operators for higher-order decompositions. Technical Report SAND2006-2081, Sandia National Laboratories, April 2006.

CP decomposition

As mentioned in the lecture that introduced tensors, one of the main difficulty when manipulating tensors is the curse of dimensionality. In an attempt to overcome this issue, the CP decomposition was created. To introduce this concept, you may use the help of the following guideline and bibliography:

Guideline

- Give the intuition behind the CP decomposition and its relation with one of the characterization of the rank of a matrix. Write the CP decomposition with the Khatri–Rao product.
- Discuss the different properties of the tensor rank and compare them to the matrix case (non-closedness, values depending on the field, maximal value, hardness to find it, ...).
- Define the border rank and explain its idea.
- Give an example that could lead to numerical cancellation.
- Discuss the following application: the relation between the CP decomposition and Strassen's algorithm, the approximation of the multidimensional Newton potential operator.

References

[1] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 56 of *Springer Series in Computational Mathematics*. Springer, Cham, second edition, 2019.

[2] Wolfgang Hackbusch. *The concept of stability in numerical mathematics*, volume 45 of *Springer Series in Computational Mathematics*. Springer, Heidelberg, 2014.

[3] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.

Tucker decomposition

In the previous lecture that introduced the CP decomposition, we saw one of the many existing decompositions of tensors that tries to overcome the curse of dimensionality. As such, the Tucker decomposition is another well-known decomposition. To introduce this concept, you may use the help of the following guideline and bibliography:

Guideline

- Give the intuition behind the Tucker decomposition and its relation with the SVD. Prove the existence of a Tucker decomposition for any tensor and discuss its unicity, especially the impact on the core tensor. Explain why it does not necessarily overcome the curse of dimensionality. Show that the CP decomposition is actually a special case of Tucker's decomposition.
- Present and prove the many similarities between the HOSVD and the SVD of a matrix.
- Give and prove the main properties of the HOSVD.
- Present the non sequentially truncated HOSVD algorithm. Prove its error bound with respect to the best Tucker approximation.
- Present the sequentially truncated HOSVD algorithm. Prove its error bound with respect to the best Tucker approximation. Explain its interest and why intuitively it should yield a better approximation than the non sequential truncated HOSVD.

References

[1] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 56 of *Springer Series in Computational Mathematics*. Springer, Cham, second edition, 2019.

[2] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

Tensor train decomposition

In the line of the previous lectures, the tensor train (TT) decomposition of a tensor is another famous decomposition. To introduce this concept, you may use the help of the following guideline and bibliography:

Guideline

- Give the definition of the TT decomposition and write a TT decomposition with the diagrammatic notation. Prove its existence and discuss its unicity.
- Present the hybrid formats (TT decomposition with the cores in Tucker's format) and present the memory complexity.
- Present the truncated TT SVD algorithm. Prove its error bound with respect to the best TT approximation.
- Present the TT rounding algorithm. Explain why it is an important algorithm when operations in the TT format are done.
- Explain how one can derive an efficient representation of d dimensional Laplace operators.

References

[1] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

Tensor train arithmetic

As shown in the previous lecture, the TT decomposition of a tensor offers an efficient way to store it as its memory complexity scales linearly with the dimension. Moreover, many operations between two tensors in TT format can also be done in linear complexity with respect to the dimension. To introduce the different techniques used to derive those operations, you may use the help of the following guideline and bibliography:

Guideline

- Show how addition and multiplication by a scalar can be done efficiently in the TT format. Provide the computational complexity.
- Show how the multidimensional contraction, Hadamard product, scalar product, and norm can be computed efficiently in the TT format. Provide the computational complexity. Use the diagrammatic notation to explain the different algorithms.
- Show how matrix-vector product can be computed efficiently in the TT format. Provide the computational complexity. Mention what happens when the TT ranks of the matrix are all equal to one.
- Present both the shuffle and Pot-RwCl algorithms. Provide their computational complexity. Quickly discuss how to improve the shuffle algorithm. Compare the obtained complexities with the one obtained for the matrix-vector product.

References

[1] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[2] Turul Dayar and M. Can Orhan. On vector-kronecker product multiplication with rectangular factors. *SIAM Journal on Scientific Computing*, 37(5):S526–S543, 2015.

Hierarchical Tucker decomposition

In the line of the previous lectures, the hierarchical Tucker decomposition (HTD) of a tensor is another famous decomposition. To introduce this concept, you may use the help of the following guideline and bibliography:

Guideline

- Give the definition of the HTD and write it with the diagrammatic notation. Give the number of inner nodes and leaves in a complete binary tree and prove it. Show how under certain conditions, this decomposition is a generalization of both the Tucker and TT decompositions. Discuss the advantages/disadvantages of the hierarchical format over the TT format.
- Define all the concepts necessary to present both the root-to-leaves and leaves-to-root truncation algorithms. Give the proof of their error bounds. Derive their complexity estimates.
- Mention the use of tensors in HTD to represent linear operators. Give the example of the representation of a descretized Laplace-like operator.

References

[1] Lars Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2029–2054, 2010.

[2] Daniel Kressner and Christine Tobler. Algorithm 941: htucker—a Matlab toolbox for tensors in hierarchical Tucker format. *ACM Trans. Math. Softw.*, 40(3), April 2014.

Turncation and orthogonalization of a tensor in HTD format

As shown in the previous lecture, the HTD of a tensor offers an efficient way to store it as its memory complexity scales linearly with the dimension. Moreover, many operations between two tensors in HTD can also be done in linear complexity with respect to the dimension. Among them, the orthogonalization and truncation of a tensor in HTD format are essential. To introduce the non-trivial algorithms to compute those operations, you may use the help of the following guideline and bibliography:

Guideline

- Show the expression of the t-matricization of a tensor already in the HTD format.
- Define the accumulated transfer tensors. Show the computational complexity to compute them.
- Prove the frame transformation formula.
- Show the algorithm to orthogonalize a tensor already in the HTD format. Show its computational complexity.
- Show how to truncate a tensor already in the HTD format using the accumulated tensors. Show the computational complexity estimate.

References

[1] Lars Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2029–2054, 2010.

Arithmetic of tensors in HTD format

As shown in the previous lecture, the HTD of a tensor offers an efficient way to store it as its memory complexity scales linearly with the dimension. Moreover, many operations between two tensors in HTD can also be done in linear complexity with respect to the dimension. To introduce the the different techniques used to derive those operations, you may use the help of the following guideline and bibliography:

Guideline

- Show how the μ-mode matrix product can be efficiently done in the HTD format.
- Show how the addition can be efficiently computed in the HTD format. Provide the computational complexity.
- Show how the multidimensional contraction, scalar product, and norm can be computed efficiently in the HTD format. Provide the computational complexity. Use the diagrammatic notation to explain the different algorithms.
- Define the reduced Gramians of a tensor in HTD and show how to compute it efficiently. Provide the computational complexity. Mention why reduced Gramians are a useful tool.
- Show how to efficiently truncate a sum of tensors in HTD. Provide the computational complexity. Explain why the naive ideas would not work very well.
- Show how to efficiently compute the elementwise multiplication of tensors in HTD. Provide the computational complexity.

References

[1] Daniel Kressner and Christine Tobler. Algorithm 941: htucker—a Matlab toolbox for tensors in hierarchical Tucker format. *ACM Trans. Math. Softw.*, 40(3), April 2014.

Alternating least squares method

We have seen the different principal decompositions of tensors and algorithms to build them. There exist many other methods to build such decompositions and the alternating least squares (ALS) method is one of them. To introduce this method, you may use the help of the following guideline and bibliography:

Guideline

- Define the pseudoinverse and show its relation with the SVD of a matrix.
- Define the least squares problem. Show how to obtain the smallest 2-norm minimizers. Explain how to generalize this solution to multilinear least squares problem.
- Present the idea of the ALS for the CP decomposition with its proofs. Present the algorithm with its computational complexity. Illustrate cases where it can go wrong and explain why we can observe such cases. Mention the idea of some techniques to improve the efficiency of ALS.
- Present the idea of Higher-order orthogonal (HOOI) iteration with its proofs. Present the HOOI algorithm with its computational complexity. Discuss the initialization and stopping criteria.
- Present the idea of ALS for the TT decomposition with its proofs. Present the algorithm with its computational complexity. Discuss the initialization and stopping criteria.

References

[1] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[2] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 56 of *Springer Series in Computational Mathematics*. Springer, Cham, second edition, 2019.

[3] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r1 ,r2 ,. . .,rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[4] Lars Grasedyck and Sebastian Krämer. Stable ALS approximation in the TT-format for rank-adaptive tensor completion. *Numer. Math.*, 143(4):855–904, 2019.

[5] Tamara G. Kolda. Multilinear operators for higher-order decompositions. Technical Report SAND2006-2081, Sandia National Laboratories, April 2006.

Cross approximation method for matrices

In the spirit of the previous lecture, another popular method used to build the different tensors decompositions that we have seen is the cross approximation method. To introduce the intuition behind, first study the matrix case. You may use the help of the following guideline and bibliography:

Guideline

- Define the skeleton approximation of a matrix and explain its interest. Prove existence error bound when a matrix A admits a low rank approximation.
- Define the volume of a matrix. Explain why the maximal volume submatrix yields a good skeleton approximation and what is the issue with it.
- Present the idea of the maximum volume algorithm with its proofs. Present the algorithm with its computational complexity.
- Define the functional skeleton of a function. Explain the idea to approximate the skeleton of a function in and link it with the previous bullet points. For this point, you don't need to prove everything, but the explanation should be detailed enough so that the link is clear.

References

[1] Mario Bebendorf. Approximation of boundary element matrices. *Numer. Math.*, 86(4):565–589, 2000.

[2] S. A. Goreinov and E. E. Tyrtyshnikov. The maximal-volume concept in approximation by low-rank matrices. In *Structured matrices in mathematics, computer science, and engineering, I (Boulder, CO, 1999)*, volume 280 of *Contemp. Math.*, pages 47–51. Amer. Math. Soc., Providence, RI, 2001.

[3] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra Appl.*, 261:1–21, 1997.

[4] S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. How to find a good submatrix. In *Matrix methods: theory, algorithms and applications*, pages 247–256. World Sci. Publ., Hackensack, NJ, 2010.

Cross approximation method for tensors

In the previous lecture, the idea of the cross approximation method for matrices was introduced. Now the goal is to generalize this method to compute decompositions of higher order tensors. To do that, you may use the help of the following guideline and bibliography:

Guideline

- Present the intuition of the TT cross approximation algorithm as in [3]. Give the computational complexity of one sweep by assuming that the column index are given.
- Explain the algorithm given in [3] that may find a submatrix of good volume. Give an example where it can go wrong. With this algorithm, give the final TT cross approximation algorithm.
- Explain the application of the TT-cross approximation algorithm to integration of high dimensional functions. Present the fast TT contraction algorithm.
- Explain the idea of adaptive cross approximation of functions of 3 and 4 variables. For this point, you don't need to prove everything, but the explanation should be detailed enough so that the idea is clear. You may also use the discussion of the approximation of functional skeleton introduced in the previous lecture.

References

[1] Mario Bebendorf. Approximation of boundary element matrices. *Numer. Math.*, 86(4):565–589, 2000.

[2] M. Bebendorf. Adaptive cross approximation of multivariate functions. *Constr. Approx.*, 34(2):149–179, 2011.

[3] Ivan Oseledets and Eugene Tyrtyshnikov. TT-cross approximation for multidimensional arrays. *Linear Algebra Appl.*, 432(1):70–88, 2010.