

abess: A Fast Best-Subset Selection Library in Python and R

Jin Zhu¹

Xueqin Wang²

Liyuan Hu¹

Junhao Huang¹

Kangkang Jiang¹

Yanhang Zhang³

Shiyun Lin⁴

Junxian Zhu⁵

ZHUJ37@MAIL2.SYSU.EDU.CN

WANGXQ20@USTC.EDU.CN

HULY5@MAIL2.SYSU.EDU.CN

HUANGJH256@MAIL2.SYSU.EDU.CN

JIANGKK3@MAIL2.SYSU.EDU.CN

ZHANGYH98@RUC.EDU.CN

SHIYUNLIN@STU.PKU.EDU.CN

JUNXIAN@NUS.EDU.SG

¹ Department of Statistical Science, Sun Yat-Sen University, Guangzhou, GD, China

² Department of Statistics and Finance/International Institute of Finance, School of Management, University of Science and Technology of China, Hefei, Anhui, China

³ School of Statistics, Renmin University of China, Beijing, China

⁴ Center for Statistical Science, Peking University, Beijing, China

⁵ Saw Swee Hock School of Public Health, National University of Singapore, Singapore

Editor: Alexandre Gramfort

Abstract

We introduce a new library named **abess** that implements a unified framework of best-subset selection for solving diverse machine learning problems, e.g., linear regression, classification, and principal component analysis. Particularly, **abess** certifiably gets the optimal solution within polynomial time with high probability under the linear model. Our efficient implementation allows **abess** to attain the solution of best-subset selection problems as fast as or even 20x faster than existing competing variable (model) selection toolboxes. Furthermore, it supports common variants like best subset of groups selection and ℓ_2 regularized best-subset selection. The core of the library is programmed in C++. For ease of use, a Python library is designed for convenient integration with **scikit-learn**, and it can be installed from the Python Package Index (PyPI). In addition, a user-friendly R library is available at the Comprehensive R Archive Network (CRAN). The source code is available at: <https://github.com/abess-team/abess>.

Keywords: best-subset selection, high-dimensional data, splicing technique

1. Introduction

Best-subset selection (BSS) is imperative in machine learning and statistics. It aims to find a minimally adequate subset of variables to accurately fit the data, naturally reflecting Occam's razor principle of simplicity. Nowadays, the BSS also has far-reaching applications in every facet of research like medicine and biology because of the surge of large-scale data sets across a variety of fields. As a benchmark optimization problem in machine learning and statistics, the BSS is also well-known as an NP-hard problem (Natarajan, 1995). However, recent progress shows that the BSS can be efficiently solved (Huang et al., 2018; Zhu et al., 2020; Gómez and Prokopyev, 2021). Especially, the ABESS algorithm using a splicing

*. Jin Zhu and Liyuan Hu contributed equally. Xueqin Wang is the corresponding author.

Learning	Target	Solver (Reference)
Supervised (model the variation of response y)	$y \in \mathbb{R}$	LinearRegression (Zhu et al., 2020)
	$y \in \{0, 1\}$	LogisticRegression (Hosmer et al., 1989)
	$y \in \{0, 1, 2, \dots\}$	PoissonRegression (Vincent and Claire, 2010)
	$y \in (0, \infty)$	GammaRegression (Vincent and Claire, 2010)
	$y \in \{\text{type1}, \text{type2}, \dots\}$	MultinomialRegression (Krishnapuram et al., 2005)
	$y \in \{\text{level1}, \text{level2}, \dots\}$	OrdinalRegression (Wurm et al., 2021)
Unsupervised	$y \in \mathbb{R} \times \{0, 1\}$	CoxPHSurvivalAnalysis (Pölsterl, 2020)
	$y \in \mathbb{R}^d$	MultiTaskRegression (Zhang and Yang, 2017)
	Dimension reduction	SparsePCA (d’Aspremont et al., 2008)
	Matrix decomposition	RobustPCA (Cai et al., 2019)

Table 1: The supported best-subset selection solvers. PCA: principal component analysis.

technique finds the best subset under the classical linear model in polynomial time with high probability (Zhu et al., 2020), making itself even more attractive to practitioners.

We present a new library named **abess** that implements a unified toolkit based on the splicing technique proposed by Zhu et al. (2020). The supported solvers in **abess** are summarized in Table 1. Furthermore, our implementation improves computational efficiency by warm-start initialization, sparse matrix support, and OpenMP parallelism. **abess** can run on most Linux distributions, Windows 32 or 64-bit, and macOS with Python (version ≥ 3.6) or R (version $\geq 3.1.0$), and can be easily installed from PyPI¹ and CRAN². **abess** provides complete documentation³, where the API reference presents the syntax and the tutorial presents comprehensible examples for new users. It relies on Github Actions⁴ for continuous integration. The PEP8/tidyverse style guide keeps the source Python/R code clean. Code quality is assessed by standard code coverage metrics (Myers et al., 2011). The coverages for the Python and the R packages at the time of writing are 97% and 96%, respectively. The source code is distributed under the GPL-3 license.

2. Architecture

Figure 1 shows the architecture of **abess**, and each building block will be described as follows. The **Data** class accepts the (sparse) tabular data from Python and R interfaces, and returns an object containing the predictors that are (optionally) screened (Fan and Lv, 2008) or normalized. The **Algorithm** class implements the generic splicing technique for the BSS with the additional support for group-structure predictors (Zhang et al., 2021), ℓ_2 -regularization for parameters (Bertsimas and Parys, 2020), and nuisance selection (Sun and Zhang, 2021). The concrete algorithms are programmed as subclasses of **Algorithm** by rewriting the virtual function interfaces of class **Algorithm**. Seven implemented BSS tasks are presented in Figure 1. Beyond that, the modularized design facilitates users to extend the library to various machine learning tasks by writing a subclass of **Algorithm**. The **Metric** class assesses the estimation returned by the **Algorithm** class by the cross-validation or information criteria like the Akaike information criterion and the high dimensional Bayesian information criterion (Akaike, 1998; Wang et al., 2013). Python and R interfaces collect and process the results of the **Algorithm** and **Metric** classes. The **abess** Python library is compatible with **scikit-learn** (Pedregosa et al., 2011). For each solver (e.g., **LinearRegression**) in **abess**, Python users can not only use a familiar **scikit-learn** API to

1. <https://pypi.org/project/abess>
 2. <https://cran.r-project.org/web/packages/abess>
 3. <https://abess.readthedocs.io> and <https://abess-team.github.io/abess>
 4. <https://github.com/abess-team/abess/actions>

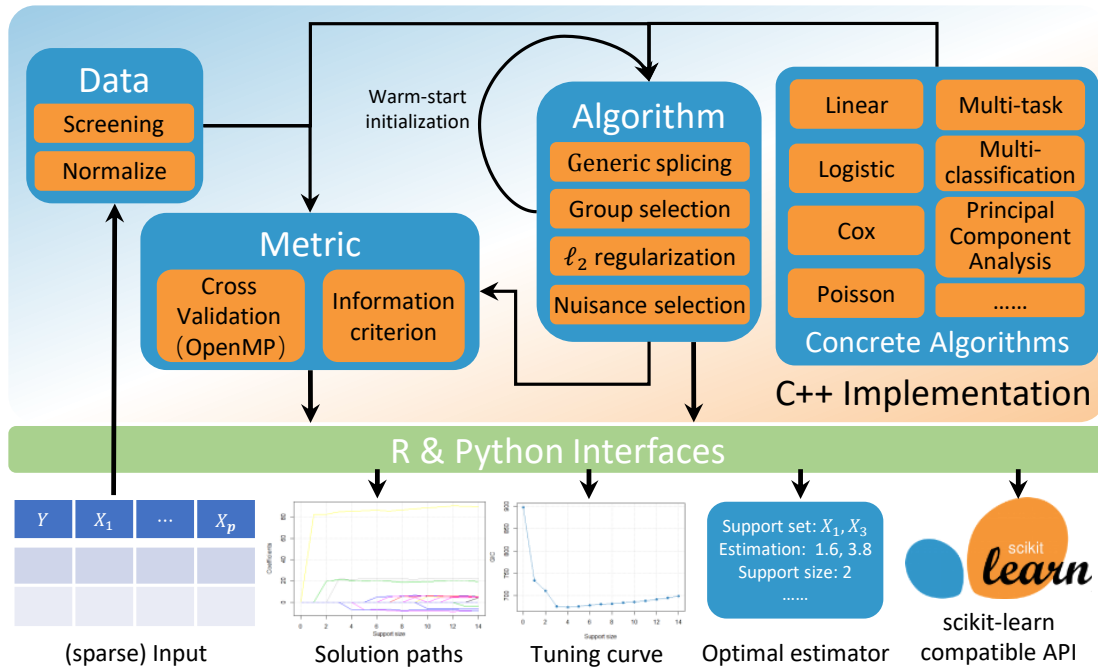


Figure 1: abess software architecture.

```

1 library(abess)
2 dat <- generate.data(n = 300, p = 1000, beta = c(3, -2, 0, 0, 2, rep(0, 995)))
3 best_est <- extract(abess(dat$x, dat$y, family = "gaussian"))
4 cat("Selected subset:", best_est$support.vars,
5     "and coefficient estimation:", round(best_est$support.beta, digits = 2))
6 ## Selected subset: x1 x2 x5 and coefficient estimation: 2.96 -2.05 1.9

```

Figure 2: Using the `abess` R library on a synthetic data set to illustrate its optimality. The data set comes from a linear model with the true sparse coefficients given by `beta`.

train the model but also easily create a `scikit-learn` pipeline including the model. In the R library, `S3` methods are programmed such that generic functions (like `print`, `coef`, and `plot`) can be directly used to attain the BSS results and visualize solution paths or tuning value curves.

3. Usage Examples

Figure 2 shows that the `abess` R library exactly selects the effective variables and accurately estimates the coefficients. Figure 3 illustrates the integration of the `abess` Python interface with `scikit-learn`'s modules to build a non-linear model for diagnosing malignant tumors. The output of the code reports the information of the polynomial features for the selected model among candidates, and its corresponding area under the curve (AUC), which is 0.966, indicating the selected model would have an admirable contribution in practice.

4. Performance

We compare `abess` with popular variable selection libraries in Python and R through regression, classification, and PCA. The libraries include: `scikit-learn` (a benchmark Python library for machine learning), `celer` (a fast Python solver for ℓ_1 -regularization optimization, Massias et al., 2018, 2020), and (an elastic-net R solver for sparse PCA, Zou et al., 2006). All computations are conducted on a Ubuntu platform with Intel(R) Core(TM) i9-9940X

```

1 from abess.linear import LogisticRegression
2 from sklearn.datasets import load_breast_cancer
3 from sklearn.pipeline import Pipeline
4 from sklearn.metrics import make_scorer, roc_auc_score
5 from sklearn.preprocessing import PolynomialFeatures
6 from sklearn.model_selection import GridSearchCV
7 # combine feature transform and model:
8 pipe = Pipeline([('poly', PolynomialFeatures(include_bias=False)),
9 ('logreg', LogisticRegression())])
10 param_grid = {'poly__interaction_only':[True, False], 'poly__degree':[1, 2, 3]}
11 # Use cross validation to tune parameters:
12 scorer = make_scorer(roc_auc_score, greater_is_better=True)
13 grid_search = GridSearchCV(pipe, param_grid, scoring=scorer, cv=5)
14 # load and fitting example data set:
15 X, y = load_breast_cancer(return_X_y=True)
16 grid_search.fit(X, y)
17 # print the best tuning parameter and associated AUC score:
18 print([grid_search.best_params_, grid_search.best_score_])
19 # >>> [{'poly__degree': 2, 'poly__interaction_only': True}, 0.9663829492654472]

```

Figure 3: Example of using the abess Python library with scikit-learn.





Library	Version	Superconductivity (3895 × 85400)			Cancer (118 × 22215)			Musk (7074 × 166)		
		MSE	NNZ	Runtime	AUC	NNZ	Runtime	AUC	NNZ	Runtime
scikit-learn (ℓ_1)	1.0.0	33.56	1126.70	1043.96	0.92	366.55	62.16	0.97	165.40	53.63
celer	0.6.1	88.58	30.00	173.25	0.91	20.65	26.24	0.97	162.15	2.00
scikit-learn (ℓ_0)	1.0.0				X	X	X	X	X	X
abess	0.4.5	41.72	81.50	110.41	0.96	1.00	2.91	0.97	155.25	140.68

Table 2: Average performance on the superconductivity data set (for regression), the cancer and the musk data sets (for classification) (Chin et al., 2006; Dua and Graff, 2017; Hamidieh, 2018) based on 20 randomly drawn test sets. NNZ: the number of non-zero elements. Runtime is measured in seconds. scikit-learn (ℓ_1): LassoCV (for regression) and LogisticRegressionCV (for classification). celer: LassoCV (for regression) and LogisticRegression (for classification). scikit-learn (ℓ_0): Orthogonal-MatchingPursuit (for regression). **X**: not available. : memory overflow.

CPU @ 3.30GHz and 48 RAM. Python version is 3.9.1 and R version is 3.6.3. Table 2 displays the regression and classification analysis results, suggesting abess derives parsimonious models that achieve competitive performance in few minutes. Particularly, for the cancer data set, it is more than 20x faster than scikit-learn (ℓ_1). The results of the sparse PCA (SPCA) are demonstrated in Table 3. Compared with elasticnet, abess consumes less than a tenth of its runtime but explains more variance under the same sparsity level.

5. Conclusion

abess is a fast and comprehensive library for solving various BSS problems with statistical guarantees. It offers user-friendly interfaces for both Python and R users, and seamlessly integrates with existing ecosystems. Therefore, the abess library is a potentially indispensable toolbox for machine learning and related applications. Future versions of abess intend to support other important machine learning tasks, and adapt to advanced machine learning pipelines in Python and R (Lang et al., 2019; Feurer et al., 2021; Binder et al., 2021).

Sparsity	5		10		20		
	Library	elasticnet	abess	elasticnet	abess	elasticnet	abess
Explained variance		1.37	2.28	1.61	3.03	2.00	3.88
Runtime (seconds)		15.87	1.06	15.77	1.07	85.54	1.05

Table 3: Performance of the SPCA when 5, 10, 20 elements in the loading vector of the first principal component are non-zero. The data set has 217 observations, each of which has 1,413 genetic factors (Christensen et al., 2009). elasticnet: version 1.3.0.

Acknowledgments

We would like to thank three reviewers for their constructive suggestions and valuable comments, which have substantially improved this article and the `abess` library. Wang’s research is partially supported by NSFC (72171216, 71921001, 71991474), The Key Research and Development Program of Guangdong, China (2019B020228001), and Science and Technology Program of Guangzhou, China (202002030129). Zhu’s research is partially supported by the Outstanding Graduate Student Innovation and Development Program of Sun Yat-Sen University (19lgyjs64). Zhang’s research is supported by the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China (22XNH161). We are grateful to UCI Machine Learning Repository for sharing the superconductivity and musk data sets.

References

- Hiroto Akaike. Information theory and an extension of the maximum likelihood principle. In *Selected Papers of Hirotugu Akaike*, pages 199–213. Springer, 1998.
- Dimitris Bertsimas and Bart Van Parys. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *The Annals of Statistics*, 48(1):300 – 323, 2020. doi: 10.1214/18-AOS1804. URL <https://doi.org/10.1214/18-AOS1804>.
- Martin Binder, Florian Pfisterer, Michel Lang, Lennart Schneider, Lars Kotthoff, and Bernd Bischl. mlr3pipelines – flexible machine learning pipelines in r. *Journal of Machine Learning Research*, 22(184):1–7, 2021. URL <http://jmlr.org/papers/v22/21-0281.html>.
- Hanqin Cai, Jianfeng Cai, and Ke Wei. Accelerated alternating projections for robust principal component analysis. *Journal of Machine Learning Research*, 20(1):685–717, 2019.
- Koei Chin, Sandy DeVries, Jane Fridlyand, Paul T Spellman, Ritu Roydasgupta, Wen-Lin Kuo, Anna Lapuk, Richard M Neve, Zuwei Qian, Tom Ryder, Fanqing Chen, Heidi Feiler, Taku Tokuyasu, Chris Kingsley, Shanaz Dairkee, Zhenhang Meng, Karen Chew, Daniel Pinkel, Ajay Jain, Britt Marie Ljung, Laura Esserman, Donna G Albertson, Frederic M Waldman, and Joe W Gray. Genomic and transcriptional aberrations linked to breast cancer pathophysiology. *Cancer Cell*, 10(6):529–541, December 2006.
- Brock C Christensen, E Andres Houseman, Carmen J Marsit, Shichun Zheng, Margaret R Wrensch, Joseph L Wiemels, Heather H Nelson, Margaret R Karagas, James F Padbury, Raphael Bueno, David J Sugarbaker, Ru-Fang Yeh, John K Wiencke, and Karl T Kelsey. Aging and Environmental Exposures Alter Tissue-Specific DNA Methylation Dependent upon CpG Island Context. *PLOS Genetics*, 5(8):e1000602, August 2009.
- Alexandre d’Aspremont, Francis Bach, and Laurent El Ghaoui. Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research*, 9(7), 2008.
- Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.

- Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5): 849–911, 2008. doi: <https://doi.org/10.1111/j.1467-9868.2008.00674.x>.
- Matthias Feurer, Jan N. van Rijn, Arlind Kadra, Pieter Gijsbers, Neeratyoy Mallik, Sahithya Ravi, Andreas Müller, Joaquin Vanschoren, and Frank Hutter. Openml-python: an extensible python api for openml. *Journal of Machine Learning Research*, 22(100):1–5, 2021. URL <http://jmlr.org/papers/v22/19-920.html>.
- Andrés Gómez and Oleg A. Prokopyev. A mixed-integer fractional optimization approach to best subset selection. *INFORMS Journal on Computing*, 33(2):551–565, 2021. doi: 10.1287/ijoc.2020.1031.
- Kam Hamidieh. A data-driven statistical model for predicting the critical temperature of a superconductor. *Computational Materials Science*, 154:346–354, 2018.
- David W. Hosmer, Borko Jovanovic, and Stanley Lemeshow. Best subsets logistic regression. *Biometrics*, 45(4):1265–1270, 1989. URL <http://www.jstor.org/stable/2531779>.
- Jian Huang, Yuling Jiao, Yanyan Liu, and Xiliang Lu. A constructive approach to l_0 penalized regression. *Journal of Machine Learning Research*, 19(1):403–439, 2018.
- Balaji Krishnapuram, Lawrence Carin, Mario A. T. Figueiredo, and Alexander J. Hartemink. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):957–968, 2005. doi: 10.1109/TPAMI.2005.127.
- Michel Lang, Martin Binder, Jakob Richter, Patrick Schratz, Florian Pfisterer, Stefan Coors, Quay Au, Giuseppe Casalicchio, Lars Kotthoff, and Bernd Bischl. mlr3: A modern object-oriented machine learning framework in R. *Journal of Open Source Software*, dec 2019. doi: 10.21105/joss.01903.
- Mathurin Massias, Alexandre Gramfort, and Joseph Salmon. Celer: a fast solver for the lasso with dual extrapolation. In *International Conference on Machine Learning*, volume 80, pages 3321–3330, 2018.
- Mathurin Massias, Samuel Vaiter, Alexandre Gramfort, and Joseph Salmon. Dual extrapolation for sparse glms. *Journal of Machine Learning Research*, 21(234):1–33, 2020. URL <http://jmlr.org/papers/v21/19-587.html>.
- Glenford J Myers, Corey Sandler, and Tom Badgett. *The Art of Software Testing*. John Wiley & Sons, 2011.
- Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995. doi: 10.1137/S0097539792240406.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher,

- Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. URL <http://jmlr.org/papers/v12/pedregosa11a.html>.
- Sebastian Pölsterl. scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, 21(212):1–6, 2020. URL <http://jmlr.org/papers/v21/20-729.html>.
- Qiang Sun and Heping Zhang. Targeted inference involving high-dimensional data using nuisance penalized regression. *Journal of the American Statistical Association*, 116(535):1472–1486, 2021. doi: 10.1080/01621459.2020.1737079.
- Calcagno Vincent and de Mazancourt Claire. glmulti: An r package for easy automated model selection with (generalized) linear models. *Journal of Statistical Software*, 34(12):1–29, 2010. doi: 10.18637/jss.v034.i12.
- Lan Wang, Yongdai Kim, and Runze Li. Calibrating nonconvex penalized regression in ultra-high dimension. *The Annals of Statistics*, 41(5):2505 – 2536, 2013. doi: 10.1214/13-AOS1159.
- Michael J. Wurm, Paul J. Rathouz, and Bret M. Hanlon. Regularized ordinal regression and the ordinalnet r package. *Journal of Statistical Software*, 99(6):1–42, 2021. doi: 10.18637/jss.v099.i06.
- Yanhang Zhang, Junxian Zhu, Jin Zhu, and Xueqin Wang. A splicing approach to best subset of groups selection. *arXiv preprint arXiv:2104.12576*, 2021.
- Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 5(1):30–43, 09 2017. ISSN 2095-5138. doi: 10.1093/nsr/nwx105.
- Junxian Zhu, Canhong Wen, Jin Zhu, Heping Zhang, and Xueqin Wang. A polynomial algorithm for best-subset selection problem. *Proceedings of the National Academy of Sciences*, 2020. doi: 10.1073/pnas.2014241117. URL <https://www.pnas.org/doi/abs/10.1073/pnas.2014241117>.
- Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006. doi: 10.1198/106186006X113430. URL <https://doi.org/10.1198/106186006X113430>.