# The Emergence of Zipf's Law: Spontaneous Encoding Optimization by Users of a Command Language

STEPHEN R. ELLIS AND ROBERT J. HITCHCOCK

*Abstract*—The distribution of commands issued by experienced users of a computer operating system allowing command customization tends to conform to Zipf's law. This result documents the emergence of a statistical property of natural language as users master an artificial language. Analysis of Zipf's law by Mandelbrot and Cherry shows that its emergence in the computer interaction of experienced users may be interpreted as evidence that these users optimize their encoding of commands. Accordingly, the extent to which users of a command language exhibit Zipf's law can provide a metric of the naturalness and efficiency with which that language is used.

## I. Introduction

THE PROCESS OF EVOLUTION has resulted in the matching of many aspects of human behavior and physiology to properties of the physical environment. Similar adaptation to the informational environment has shaped the evolution of natural human language. The following cross-sectional study shows how this latter type of adaptation manifests itself in the use of a command language.

One of the striking features of coherent natural language text is the novelty of the sentences it contains. The probability that a reader has previously encountered any sentence of average length is extremely small [4]. Nevertheless, the uniqueness of such sentences does not rule out other regularities in natural language text. For example, a particular language such as English exhibits measurable conditional probabilities describing the likelihood that a particular letter will follow another. If these probabilities are based on sufficiently long preceding strings of letters and spaces, they can be used with Monte Carlo methods to approximate syntacticly correct, albeit meaningless English sentences [7]. These probabilities can, in fact, be used to determine authorship of texts since they can reflect unique properties such as the frequency of word or phrase usage [6].

One of the factors influencing the development of idiosyncratic usage patterns by users is the frequency with which particular ideas or objects are discussed. Particularly when these require many words for denotation, jargonish abbreviations are created as sometimes cryptic substitutes. Government agencies are particularly adept at this practice. A variety of methods of abbreviation may be used [11], but in all cases the abbreviator must trade off conciseness against uniqness and general readability [5].

Shortening of the words for frequently used concepts is evident as a general feature of language. The more frequently used words tend to be shorter than those used less often. Partially as a consequence of this shortening and of the pattern of abbreviation [8], the distribution of the frequency of word use in any moderately large sample of natural language, e.g., 1500 words, exhibits an empirical relation known as Zipf's law [3], [16].

If $p(n_l)$ is the distribution of frequency of use for all words $l$ and if $n_l$ is the rank order of the frequency of use so that the most frequent word is ranked first, the next most frequent ranked second, the next third, [redacted], then Zipf's law is

$$p(n_l) = Pn_l^{-1} \tag{1}$$

where $P$ is proportional to the number of words in the text.

This relation is found in the text of all alphabetic human languages and is thought to result from the adaptation of the languages to the limits of the communication channels between speakers [12]. The adaptation is possible because natural human language is inherently modifiable. In contrast, command languages that have arisen for control of computers generally do not have this flexibility. Users must learn the commands and syntax, but they generally have very limited ability to modify either.

Consequently, considerable interest has focused on techniques for the design of command languages for dialogues between men and computer-based machines [2], [11]. This design process is especially difficult because the needs and abilities of experienced users differ from those of novices [15]. Since designers have recognized that users can help define a good language, some command languages are intended to be customized. The adaptability of these languages gives them some characteristics of natural language. Accordingly, one might expect statistical properties of natural language such as Zipf's law to emerge in command languages as their users become more adept.

The emergence of Zipf's law in a command language not designed to conform to it would be interesting because Mandelbrot [12] has attributed the law to a constrained optimization of transmission of information between users.

Manuscript received February 12, 1985; revised September 27, 1985.
The authors are with the NASA Ames Research Center, Moffet Field, CA 94035, USA.
IEEE Log Number 8608399.

0018-9472/86/0500-0423$01.00 ©1986 IEEE

Since the constraint is thought to arise from limitations on usable word length, the extent to which a user's command-language text sample exhibited it would be a measure of how efficiently he was using the language to encode his commands.

In Mandelbrot's general analysis, he has shown that for communication systems in which the average cost per unit of information is minimized and for which the encoding cost per word is considered proportional to word length, the following frequency distribution holds:

$$p(n_l) = P(n_l + m)^{-B}. \tag{2}$$

$P$, $m$, and $B$ are derived positive constants and $p(n_l)$ and $n_l$ are as before the distribution of word frequencies and word rank order, respectively. Zipf's law is a special case where $m = 0$ and $B = 1$.

Cherry [3] has developed a less general analysis showing Zipf's law to be a consequence of an optimal selection of word lengths and of the fact that where frequency is a decreasing function of word length, the length of a word is approximated by its log rank order.[1]

If command strings are considered words, Mandelbrot and Cherry's analyses imply that deviations from Mandelbrot's generalization of Zipf's law would be evidence of nonoptimal command encoding. Accordingly, analysis of collections of commands generated by users with different experience levels ought to reflect their respective skill levels by the extent to which their communications exhibit Zipf's law. Specifically, if a subject's frequency-of-use data is fit by the Mandelbrot generalized function, the expectation is that: as the skill level of the user increases, the fit of the theoretical function should improve, the exponent $B$ should approach each subject's corresponding value for natural text, and the constant $m$ should approach zero.

## II. Methods

We have tested this hypothesis with a cross-sectional study of users' histories of interaction with the Unix operating system. The ten subjects in the study used a user-modifiable command interpreter called the *C-shell* [9] while working at the NASA Ames Research Center.

Subjects varied from systems programmers to novices. Samples of their interaction with the computer were collected unobtrusively each time they logged on during a period of several months (totals = 1500–2500 words). The text created by their use of the command shell interpreter was matched (within 12 percent) for the number of words with text generated by each subject in the course of his word processing on the same computer system.

Words were defined as alphanumeric strings separable by whitespace, excluding delimiters such as quotes and parentheses. All character strings in the computer-interaction text significant to the command interpreter were acceptable words. The BMDP PAR nonlinear regression program was used to fit Mandelbrot's generalized function to each subject's frequency-of-use data by minimizing the mean square deviation. The stability of these fits was tested by trying a variety of initial parameter estimates, and resulting values proved robust to these variations. As seen from the illustrative data in Fig. 1, the fits were quite good.

The subject's relative familiarity with the shell command language was measured in various ways: a subjective ranking, which was the basis of subject selection; counts of the number of different standard shell commands present in the sample of interaction; and a count of the total number of unique strings present in the sample, i.e., vocabulary size. All these measures were highly intercorrelated ($r > 0.71$, $df = 8$, $p < 0.03$); however, the last seemed the most comprehensive and objective. Simply counting the number of standard commands executed by a user ignores the use of modifying flags and the variety possible in specifying file names. Both of these elements are important parts of a user's contact with the command interpreter.

## III. Results

Table I presents a summary of the parameter estimates and other characteristics of the natural and command text collected from each subject. The values in this table correspond to those for the nonlinear regression run using (2) as a model. Especially evident from this table is the difference between the two types of text. The power parameter for the ordinary text is consistently a bit greater than the expected $-1$; the fits are consistently better than those to the command text; and the $m$ parameter, which is related to lack of linearity in the fitted function, is consistently close to zero.

More significantly, if the command text's mean square measure of goodness of fit and its two parameters not directly related to text sample size, $m$ and $B$, are used as independent variables in a multiple regression to predict the relative user sophistication measured by vocabulary size, 0.952 was the multiple $r$ of the regression.[2,3] This regression confirms the hypothesis that the experienced users have command texts more clearly exhibiting Zipf's law. Fig. 1 illustrates this confirmation for an experienced and inexperienced user.

We have examined alternative interpretations of this regression. For example, individual differences might be confounded with experience. Thus the Zipf-like behavior

[1] Where $M$ is the alphabet size in characters, $l$ is word length, and $n_l$ is rank order: $l \sim l_0 + \log_M(n_l + 1/(M+1))$. However, $1/(M+1)$ may be neglected since $M$ is usually greater than 25. Since encoding cost $c_l$ is proportional to $l$, $c_l \sim c_{l0} + \log_M(n_l)$. Finally, the optimal selection of word length can be shown to occur when $p(n_l)$ is of the form $p(n_l) = Qe^{-kc_l}$. By substitution for $c_l$, Zipf's law is a direct consequence of these last two equations.

[2] Overall: $F = 19.46$, $df = 3, 6$, $p < 0.002$. Mean square measure of fit: $F = 6.36$, $df = 1, 6$, $p < 0.05$. Parameter $m$: $F = 6.28$, $df = 1, 6$, $p < 0.05$. Parameter $B$: $F = 19.8$, $df = 1, 6$, $p < 0.005$.

[3] Studies of frequency of across-subject Unix command use show that the frequency distribution does not exhibit Zipf's law. When these data [11] are fit by our methods to Mandelbrot's generalized equation, they show substantial deviation from the Zipf case: $p(n_l) = 10^{12}(n_l + 27.01)^{-6.147}$. The use of more single-character codes for frequently used commands would allow closer approximation of the optimal Zipf case.

TABLE I
PARAMETER ESTIMATES AND MEASURES OF GOODNESS[1]

| Subject | $P$ | Parameters ($m$) | $-B$ | Mean Square | Unique Words[2] | Standard Unix Commands[2] |
|---|---|---|---|---|---|---|
| | | | Natural Text | | | |
| jaw | 242.88 | 0.2307 | −0.8789 | 0.569 | 1527 | |
| how | 182.41 | −0.2069 | −0.7507 | 1.922 | 745 | |
| sil | 393.28 | −0.0206 | −0.8376 | 1.442 | 1201 | |
| ren | 223.26 | −0.1116 | −0.8014 | 0.649 | 959 | |
| bob | 141.23 | −0.6350 | −0.7320 | 2.060 | 677 | |
| mcg | 232.18 | −0.4664 | −0.7917 | 1.922 | 687 | |
| hit | 411.10 | 0.9001 | −0.9398 | 0.952 | 1098 | |
| rob | 442.80 | 0.5048 | −0.9041 | 2.859 | 1385 | |
| pie | 258.77 | 0.5543 | −0.8683 | 0.748 | 618 | |
| mar | 147.89 | 1.0150 | −0.8894 | 0.783 | 468 | |
| | | | Command Text | | | |
| jaw | 435.6 | 0.37 | −0.944 | 1.84 | 536 | 81 |
| how | 933.5 | 4.53 | −1.077 | 4.38 | 463 | 68 |
| sil | 8312.5 | 10.71 | −1.438 | 6.17 | 600 | 67 |
| ren | 2050.3 | 4.44 | −1.280 | 5.97 | 420 | 56 |
| bob | 11175.7 | 7.45 | −1.760 | 11.52 | 261 | 48 |
| mcg | 433.7 | 1.28 | −0.879 | 12.20 | 457 | 42 |
| hit | 974.7 | 2.67 | −1.051 | 16.67 | 292 | 42 |
| rob | 11219.5 | 8.56 | −1.578 | 9.14 | 410 | 38 |
| pie | 70678.0 | 8.66 | −2.350 | 18.02 | 133 | 21 |
| mar | 143474.0 | 12.76 | −2.630 | 9.80 | 124 | 20 |

[1] Produced by regression using Mandelbrot's generalized equation as a model for frequency-of-use data from natural and command texts.
[2] Represent measures of the subjects' experience on the computer system: total vocabulary measured by number of unique strings used and total number of standard UNIX commands used. Subjects are listed in decreasing order of experience.

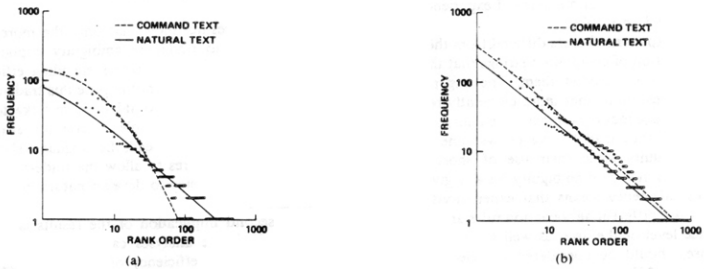


Fig. 1. Comparison of frequency-of-use data from an experienced (jaw) and inexperienced (mar) computer user. Data from both natural and command texts are drawn on a Zipf plot of frequency of use against rank order frequency.

found in the command text might not be attributable to more efficient use of the command language but rather to more efficient use of language in general. This possibility is discounted by the essentially zero within-subject correlation between each subject and his parameter estimates for normal and command text ($r^2 \ll .032$ for all comparisons). Why then do the experienced users exhibit the more classic Zipf functions in the command data?

## IV. DISCUSSION

Our partial explanation is that the command language itself is designed to produce it; the Zipf function only becomes evident when a sufficient variety of commands are known. This explanation would require the designer to have *a priori* estimates of the frequency of use of the commands in general use. Examination of the command names used by Unix shows that something like this did take place, though it appears to have been incomplete [14]. Commands likely to be frequently used, such a *ls*, which lists a directory, are only two characters whereas commands likely to be less frequent, such as *mkdir*, which makes new directories, are longer. However, there are very few single-character commands (less than two percent), and those that are used do not generally correspond to the most frequently needed functions [9], [10].

Our users of Unix, however, were not restricted to use the command names provided by the basic system. Fea-

tures of the operating system and its command interpreter allow the abbreviation of frequently used strings or the creation of new commands by concatenation of standard ones [9]. The user could in fact use the *alias* command to assign any non-null string as an abbreviation for any command or command sequence. Thus there was no constraint on the type of abbreviation that could be selected. Typically, users' abbreviations were truncations of existing commands and cryptic strings of consonants that appear to have been produced by vowel deletion. Thus the *alias*-ing feature allowed the user to customize his encoding of commands to reflect his pattern of use. It thus would encourage development of more typical Zipf functions in the command data of experienced users.

The way this capability affected the user's encoding is surprising: The more experienced users do not have more customized strings among their most frequent commands. On the contrary, the number of customized strings of the 25 most frequent commands for each subject negatively correlated with the experience measures of the user ($r < -0.857$, $df = 8$, $p < .01$). The customized strings of the novices, however, were not commands but mostly file names and command flags. The key observation is that the lengths of the strings differed. The average length of the most experienced user's customized strings was 2.5 characters while the least experienced average length was 5.5 characters. Furthermore, customized string length negatively correlated with all measures of experience level ($r > -0.747$, $df = 8$, $p < .02$).

Thus the experienced users differed from the novices not in the proportion of customized strings that they used, but in the way they encoded them. The experienced users adopted abbreviations that more efficiently reflected their variation in frequency of command use than did the novices. The trade-off they made, however, was one of efficiency against ambiguity since their use of shorter command names implies increased ambiguity for any given abbreviation. This difference means that experienced and novice users may use different abbreviation rules and thus that the experience level of the user, as well as frequency of command use, should be considered in selecting command abbreviations [11].

In particular, the selection of abbreviations by experienced users appears to be more influenced by the frequency of use than by other reported influences such as word length [1], [8]. This feature of their abbreviation may reflect the fact that they decode or encode their frequently used commands by direct association with the meaning and, at least for these frequent commands, do not use or benefit from encoding or decoding algorithms such as vowel deletion or truncation.

Since this was a cross-sectional study, little information can be directly gathered about the development of Zipf's law. Differences in intensity and continuity of use of the shell interpreter by the subjects preclude good estimates of total time of Unix use. Longitudinal studies will provide better information regarding the time course of emergence of Zipf's law or other statistical regularities in use of command interpreters. In addition to insight into the evolution of statistical structures in languages, measurement of the development of Zipf's law in command languages can provide indices of the naturalness and encoding efficiency of the artificial languages themselves.

The spontaneous emergence of Zipf's law in an artificial language is evidence that its human users are optimizing their rate-of-information transmission and has two practical design implications for artificial languages.

The first implication is a design guideline. Command names should be selectable in accordance with the overall expected frequency of use so that Zipf's law is satisfied. This condition can arise provided that there is an approximately exponential relationship between the number of characters per command and frequency of use. This approximation will require that the user be encouraged to use more single-character commands than is customary for usual command languages.

Though this feature of the language would result in more ambiguous potentially difficult-to-decode commands, the results reported above suggest that the cost incurred by shorter more cryptic abbreviations appears for the experienced users to be out weighed by the benefits of improved encoding efficiency. As in natural language, the difficulties in recalling many cryptic short commands can be allieviated by their frequency of use, which helps reinforce their meaning by making their encoding and decoding an overlearned skill.

Since the results show that only the more experienced users appear to trade the ambiguity implicit in shorter commands against their better encoding efficiency, command languages should not impose this trade-off on beginning users. Indeed, it would be a mistake to make efficiency of encoding the sole criteria for evaluation of a command language. Instead, the languages should incorporate *alias*-ing features to allow the trade-off between ambiguity and efficiency to develop naturally as user experience grows.

The second implication of the results is an evaluation metric. Zipf-type analyses can be used to evaluate the overall encoding efficiency of users of a particular language. By Mandelbrot and Cherry's analyses, the classic Zipf function is evidence for optimized encoding efficiency, which can translate into a maximized rate of command issuance provided the intercommand pauses are roughly uniform. Accordingly, estimation of the parameters of Mandelbrot's generalized equation, as illustrated by the results reported herein, can be used to rank the encoding efficiency of individual users or of the languages themselves. Such rankings could be used to assist the individual operators in developing more personalized efficient use of the command languages.

## References

[1] I. Benbasat and Y. Wand, "Command abbreviation behavior in human—computer interaction," *Commun. ACM*, vol. 27, no. 4, pp. 376–382, April 1984.

[2] S. K. Card, T. P. Moran, and A. Newell, *The Psychology of Human–Computer Interaction*. Hillsboro, NJ: Lawrence Erlbaum, 1983.

[3] C. Cherry, *On Human Communication, 3rd Ed.* Cambridge, MA: M.I.T. Press, 1978.

[4] N. Chomsky, *Syntactic Structures*. Paris, France: Mouton, 1968.

[5] K. D. Eason, "Dialogue design implications of task allocation between man and computer," *Ergonomics*, vol. 23, pp. 881–891, 1980.

[6] W. Fucks, "On mathematical analysis of style," *Biometrika*, vol. 39, p. 122, 1952.

[7] B. Hayes "Computational recreations," *Sci. Amer.*, vol. 249, pp. 18–28, 1983.

[8] M. H. Hodge and F. M. Pennington, "Some studies of word abbreviation behavior." *J. Exp. Psych.*, vol. 98, pp. 350–361, 1973.

[9] W. Joy, in *UNIX Programmers Manual Vol. 2B*, Computer Sciences Div., U.C. Berkeley, Berkeley, California, 1979; and in *UNIX Programmers Manual, 7th Ed.*, Bell Telephone Lab., Inc., Murray Hill, NJ, 1979.

[10] R. E. Kraut, S. Hanson, and J. F. Farber, "Command use and interface design," in *Proceedings of SIGCHI*, ACM 0-89791, pp. 120–124, 1983.

[11] L. A. Streeter, J. M. Ackroff, and G. A. Taylor, "On abbreviating command names." *Bell Syst. Tech. J.*, vol. 62, no. 60, pp. 1807–1826, 1983.

[12] B. Mandelbrot, "An information theory of the structure of language based on the theory of statistical matching of messages and coding," in *Proc. Symp. Applications of Commun. Theory*, London, Butterworth's Scientific Publications, 1953.

[13] B. Mandelbrot, *The Fractal Geometry of Nature*. San Franciso: W. H. Freeman, 1982.

[14] D. Norman, "The trouble about UNIX," *Datamation*, vol. 27, no. 12, pp. 139–150, 1981.

[15] D. Scapin, "Computer commands in restricted natural language: some aspects of memory and experience." *Human Factors*, vol. 23, pp. 365–375, 1981.

[16] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Cambridge, MA: Addison-Wesley, 1949.