ISSN 0915-3640

# PUBLICATIONS
## OF THE
# NATIONAL ASTRONOMICAL OBSERVATORY
# OF JAPAN

**Volume 8, Numbers 1–4**

---

# 国 立 天 文 台 欧 文 報 告

第 8 巻　　第 1–4 号

**MITAKA, TOKYO**

**2005**

Publications of the National Astronomical Observatory of Japan
Volume 8, Numbers 1–4

---

# CONTENTS

The Second Kiso Survey for Ultraviolet-Excess Galaxies. V
···Nagako MIYAUCHI-ISOBE and Hideo MAEHARA 1

Inaccuracies of Trigonometric Functions in Computer Mathematical Libraries
···Takashi ITO and Sadamu KOJIMA 17

Publ. Natl. Astron. Obs. Japan Vol. 8. 1–15 (2005)

# The Second Kiso Survey for Ultraviolet-Excess Galaxies. V

Nagako MIYAUCHI-ISOBE and Hideo MAEHARA

(Received March 25, 2005)

**Abstract**

The catalogue list and the identification chart of ultraviolet (UV)-excess galaxies, which have been detected on two or three-color Kiso Schmidt plates, are presented for 10 Schmidt fields. Catalogued are 127 objects, down to the photographic magnitude ～ 17.5 in the sky area of some 300 square degrees. The number of KUGs detected in this paper is much smaller than that of the high galactic area, and the total number of KUGs newly detected in the second survey reaches up to 1,954.

Key words: Ultraviolet-excess galaxies, KUGs, Survey with Schmidt telescope.

## 1. Introduction

We have been continuing the second survey of ultraviolet (UV)-excess galaxies with the Kiso 105-cm Schmidt telescope. This is a continuation and an extension of the original survey for Kiso UV-excess galaxies (KUGs) carried out by Takase and Miyauchi-Isobe (1984-1993a). Its comprehensive catalogue was published by Takase and Miyauchi-Isobe (1993b), where 8,104 KUGs were included in the covered sky area of some 5,100 square degrees. (The data of the area A0432 must be replaced to Myauchi-Isobe et al. 1997.) A variety of faint UV-excess galaxies were catalogued down to $V$～17 mag in the first series of the survey.

The main area of the KUG survey is the belt spread along $l$=180 ° from the north galactic pole toward the south. The isolated areas are those of specially selected ones relating to voids, clusters, or fields, which were studied in the previous papers (cf. Fig.II-2 of Miyauchi-Isobe and Maehara 2000). Accordingly, the sky areas treated in the second survey mainly consist of remaining fields of the main belt and isolated ones with plates of good quality. The main sky areas catalogued in this paper are 10 Schmidt fields along the galactic longitude of $l$ ～180 ° in the south hemisphere.

In the course of follow-up observations of KUGs (e.g., Maehara et al. 1987, Comte et al. 1994, Tomita et al. 1997), it is clarified that the majority of them are spiral or irregular galaxies with intense star formation in their nuclei, bars, disks, or outer regions. These samples give us clues to the understanding of triggering mechanism of star formation, and of the evolution of some types of galaxies. In addition, Seyferts, LINERs, and active galaxies with some peculiarities are minor constituents of the catalogue. Thus it is a fainter extension of the catalogue of Markarian galaxies (MKGs). In these circumstances, it is worth continuing and supplementing the first KUG survey, and we have been making the second survey (Miyauchi-Isobe and Maehara 1998, 2000, 2002, 2003).

The method of the second survey is, in principle, the same as that of the first one; $U$ (ultraviolet) and $R$ (red) double exposure 103a-E plates are used for the detection of

Table V-1. The Data of Plates.

| Area No. | Plate No. | Observation Date | Plate Center R.A. (1950.0) | Dec. | $l$ | $b$ | No. of KUGs |
|---|---|---|---|---|---|---|---|
| | | | h m | ° | ° | ° | |
| A0465 | KL6472 | 1990 Nov. 13 | 3 20 | +30 | 158 | –22 | 4 |
| A0755 | KL2383 | 1979 Oct. 30 | 4 00 | +10 | 181 | –31 | 1 |
| A0827 | KL2574 | 1980 Jan. 22 | 4 00 | + 5 | 185 | –34 | 3 |
| A1109 | KL3441 | 1981 Nov. 21 | 2 00 | –15 | 180 | –69 | 28 |
| A1177 | KL1918 | 1978 Dec. 22 | 0 40 | –20 | 107 | –82 | 38 |
| A1178 | KL1913 | 1978 Dec. 21 | 1 00 | –20 | 142 | –82 | 12(1) |
| A1179 | KL2939 | 1980 Nov. 3 | 1 20 | –20 | 168 | –80 | 20 |
| A1180 | KL4168 | 1983 Oct. 29 | 1 40 | –20 | 183 | –76 | 5 |
| A1250 | KL4137 | 1983 Sep. 12 | 1 00 | –25 | 169 | –87 | 3 |
| A1251 | KL2498 | 1979 Dec. 16 | 1 20 | –25 | 196 | –83 | 13(1) |
| | | | | | | | Total 127(2)* |

* Parenthesized is the number of duplicated objects which are doubly listed in the present survey.

KUGs. Exposure times being so set that the *U* and *R* images of early A-type stars are equally bright, the object whose *U* image is brighter than the *R* image is regarded to be bluer than early A-type stars. Typically, a field has several to ten those stars for the comparison. We pick up those galaxies as Kiso UV-excess galaxies (abbreviated as KUGs) with the visual inspection of the plate, and list their parameters in this paper. In some cases, a highly blue portion (e.g., knot, clump, shell, or ring) exists on the less blue main galaxy body. In this circumstance, the degree of UV-excess of a galaxy is estimated on the comparison of the *integrated U* and *R* brightness of the whole galaxy image on the plate, and the redder galaxy is discarded from the list.

The position, the brightness, and the morphological type of a KUG are estimated by referring to the object identified in the Palomar Sky Survey Print (PSS). Its degree of UV-excess is also confirmed by the comparison of the B (103aO) and R (103aE) print of the PSS. As a result, 127 KUGs are detected in the sky area of some 300 square degrees. The data of the sky area, photographic plate, and the number of detected objects in this work are listed in Table V-1.

## 2. Survey Catalogue

The list of detected objects and their identification charts are respectively given in Table V-2 and Figure V-1.

The evaluation procedures of detected objects, which are presented in Table V-2, are the same as those of the first survey.

Column 1: The running number according to the right ascension.

Column 2: The KUG-name composed of the values of right ascension and declination.

Column 3 and 4: The right ascension and declination for the epoch 1950.0.

Column 5: The morphological type adopted in this work is different from the traditional morphological classification, because there exist conspicuous blue (UV-excess) portions on these KUGs. Thus we adopt another classification scheme, which pays attention to the blue structures on the galaxy images (Takase et al. 1983); it consists of seven types as follows;

Ic : Irregular with blue clumps
Ig : Irregular with a giant clump
Pi : Pair of interacting components
Pd : Pair of detached components
Sk : Spiral with blue knots on the disk
Sp : Spiral with blue bar and/or nucleus
C : Compact.

The type is assigned through visual inspections of both Kiso plates and blue and red PSS prints. A colon ( : ) is attached to the type, when the type is not certainly assigned, and a question mark ( ? ) means unclassifiable.

Column 6: The image size (along the major and the minor axis) in minutes of arc on the blue PSS print.

Column 7: The apparent (blue) magnitude, which is eye-estimated on the PSS blue print referring to the known magnitude of the catalogued objects. It is usually calibrated using Zwicky catalogues, and extended towards fainter objects.

Column 8: The degree of UV-excess estimated from Kiso plates. H, M, and L denote high, medium, and low degree, respectively. Further explanation on the UV-excess is referred to Takase et al. (1983).

Column 9: The names given in previous catalogues. The abbreviated notations used in this paper have the following correspondence to those adopted in MOL (abbreviation of the catalogue list compiled by Dixon and Sonneborn 1980).

A: ARP, H: HARO, I:IC, M: MCG, MK: MKG, N: RNGC, U: UGC, V: VV, Z: ZWG, *n*Z: *n*ZW (*n*=1,2,....8), K: KUG (the previous KUG survey), and KE: KUG errata (Miyauchi-Isobe et al. 1997).

According to the identification with the other catalogues, many objects have been listed before. Especially, a number of KUGs appear in the Zwicky catalogues, and bright KUGs are identified as Markarian galaxies. There are morphologically peculiar KUGs, which appear in the MCG catalogue.

In all, the number of KUGs detected in this survey reaches up to 1,954, and the total number of KUGs in the first and the second surveys exceeds 10,000.

## 3. Discussion

The UV-excess is one of the major indices to detect active galaxies with conventional ground-based telescope. A number of Schmidt surveys have been carried out in the similar methods to us whose representative is the comprehensive work by Markarian et al. (1989). Even recently, a number of investigators have carried out new deep surveys for those objects applying the modern digitization machines and techniques treating large Schmidt plates; the Montreal survey (Coziol et al. 1993, 1994), the Madrid survey (Zamorano et al. 1994, Gallego et al. 1995), the Hamburg survey (Hopp et al. 1995, Popescu et al. 1996), and the Marseille survey (Surace and Comte 1998). According to them, major constituents of their surveys are galaxies with intense star formation (starburst) activity and/or non-thermal Seyfert-like nuclear phenomena.

The image quality and the limiting magnitude of Kiso Schmidt plates are generally less homogeneous due to the average observation condition of the site. Thus we select the plates of relatively good quality, and apply the visual (non-automatic) inspection method in order to cancel the inhomogeneity originated from the standardized inspection technique. Furthermore, we scrutinize detected objects by referring Palomar Sky Survey (PSS) prints, preventing the degradation of our survey. Since we usually pick up such objects that possess distinct blue knots, clumps, or components, we possibly miss UV-excess objects with smooth light distribution or uncertain morphological types. Therefore, we try to pick up carefully such objects according to the total color as well.

During our scrutinizing individual objects to estimate the brightness, morphological type, and degree of UV-excess, we pick up some KUGs, which exhibit other kinds of peculiar morphologies. The peculiarities are noticed in the

supplements to tableV-2 “notes on individual galaxies”, e.g., galaxies with blue clumps and knots, disk galaxies with thick and patchy arms, and sagittate, eye-, or boomerang-shaped objects. We find blue ring-like structures surrounding the nuclei of KUG0354+083 and 0402+042.

KUGs tend to be situated in pairs, groups and/or clusters of galaxies, rather than isolated galaxies of the same morphological type (e.g., Takase 1980). Actually, we have detected a conspicuous concentration of KUGs in the second survey (Miyauchi-Isobe and Maehara 1998). In this paper, we notice an interesting KUG concentration in the area A1177, as illustrated in figure 2. It is identified as SCG0045-2043 in an automated survey of southern compact groups of galaxies by Iovino (2002), which possess five faint galaxies nearly on line. Among the components, four galaxies are detected as KUGs with low, or medium degree of UV-excess. They show some peculiarities of their morphology, such as thick, or patchy arms. These groups of galaxies as well as Hickson compact groups of galaxies (HCGs) are excellent samples for the study of the influence of interaction on KUGs. Thus it is worth observing in more detail these objects to clarify the activity and the star formation of galaxies.

The authors are very much grateful to Prof. B. Takase on the continuation of the KUG survey. We are also grateful to Dr. A. Tomita of Wakayama University who gives us various suggestions on the properties of KUGs. We are grateful to the staff of Kiso Observatory for their help in observation, measurement and data processing, especially to Mr. T. Soyano for his sincere help in measurement and data processing.

## References

Comte, G., Augarde, R., Chalabaev, A., Kunth, D., and Maehara, H. 1994, “Spectrographic Study of a Large Sample of Kiso Ultraviolet-Excess Galaxies. II. Discussion”, *Astron. Astrophys*., **285**, 1-18.

Coziol, R., Demers, S., Pena, M., Torres-Peimbert, S., Fontaine, G.,Wesemael, F., Lamontagne, R. 1993, “MBG02223-1922: a Newly Identified Luminous Seyfert Galaxies”, *Mon. Not. Royal Astron. Soc*., **261**,170-174.

Coziol, R., Demers, S., Pena, M., Barneoud, R. 1994, “The Montreal Blue Galaxy Survey: II. Second List of UV-bright Candidates”, *Astron. J*., **108**, 405-413.

de Vaucouleurs, G., de Vaucouleurs, A., Corwin, Jr., H. G., Buta, R. J.,Paturel, G., and Fouque, P., 1991, *Third Reference Catalogue of Bright Galaxies*, Springer-Verlag.

Dixon, R., and Sonneborn, G., 1980, *A Master List of Nonstellar Optical Astronomical Objects*, Ohio State Univ. Press.

Gallego, J., Zamorano, J., Aragon-Salamanca, A., and Rego, M. 1995, “The Current Star Formation Rate of the Local Universe”, *Astrophys. J.*, **455**, L1-L4.

Hopp, U., Kuhn, B., Thiele, U., Birkle, K., Elsasser, H., and Kovachev, B. 1995, “A Redshift Survey for Faint Galaxies towards Voids of Galaxies”, *Astron. Astrophys. Suppl*., **109**, 537-549.

Iovino, A., 2002, “Detecting Fainter Compact Groups: Results from a New Automated Algorithm”, *Astron. J*., **285**, 2471-2489.

Maehara H., Noguchi, T., Takase, B., and Handa, T., 1987, “Spectroscopic Analysis of Kiso Ultraviolet-Excess Galaxies”, *Publ. Astron. Soc. Japan*, **39**, 393-409.

Markarian, B. E., Lipovetsukii, V.A., Stepanian, Dzh., Erastova, L. K., and Shapovalova, A. I. 1989, “The First Byurakan Survey – a Catalogue of Galaxies with Ultraviolet Continuum”, *Comm. Special Astrophys. Obs*., No. 62.

Miyauchi-Isobe, N., and Maehara, H., 1998, “The Second Kiso Survey for Ultraviolet-Excess Galaxies. I”, *Publ. Natl. Astron. Obs. Japan*, **5**, 75-97 (KUGC 2nd-I).

Miyauchi-Isobe, N., and Maehara, H., 2000, “The Second Kiso Survey for Ultraviolet-Excess Galaxies. II”, *Publ. Natl. Astron. Obs. Japan*, **6**, 1-39 (KUGC 2nd-II).

Miyauchi-Isobe, N., and Maehara, H., 2002, “The Second Kiso Survey for Ultraviolet-Excess Galaxies. III”, *Publ. Natl. Astron. Obs. Japan*, **6**, 107-146 (KUGC 2nd-III).

Miyauchi-Isobe, N., and Maehara, H., 2003, “The Second Kiso Survey for Ultraviolet-Excess Galaxies. IV”, *Publ. Natl. Astron. Obs. Japan*, **7**, 37-52 (KUGC 2nd-IV).

Miyauchi-Isobe, N., Takase, B., and Maehara, H., 1997, “Erratum: Kiso Survey for Ultraviolet-Excess Galaxies”, *Publ. Natl. Astron. Obs. Japan*, **3**, 153-158.

Popescu, C., Hopp, U., Hagen, H. J., Elsasser, H. 1996, “Search for Emission-line Galaxies towards Nearby Voids”, *Astron. Astrophys. Suppl*., **116**, 43-74.

Surace, C., and Comte, G. 1998, “The Marseille Schmidt Survey for Active Star-forming Galaxies”, *Astron. Astrophys. Suppl*., **133**, 171-179.

Takase B., 1980, “Counts of Ultraviolet-Bright Galaxies and Their Distributions in Clusters of Galaxies”, *Publ. Astron. Soc. Japan*, **32**, 605-612.

Takase B., and Miyauchi-Isobe, N., 1984, “Kiso Survey for Ultraviolet-Excess Galaxies I”, *Ann. Tokyo Astron. Obs*., 2nd Ser.,**19**, 595-638 (KUGC I).

Takase B., and Miyauchi-Isobe, N., 1985a, “Kiso Survey for Ultraviolet-Excess Galaxies II”, *Ann. Tokyo Astron. Obs*., 2nd Ser., **20**, 237-281 (KUGC II).

Takase B., and Miyauchi-Isobe, N., 1985b, “Kiso Survey for Ultraviolet-Excess Galaxies III”, *Ann. Tokyo Astron. Obs*., 2nd Ser., **20**, 335-392 (KUGC III).

Takase B., and Miyauchi-Isobe, N., 1986a, “Kiso Survey for Ultraviolet-Excess Galaxies IV”, *Ann. Tokyo Astron. Obs*., 2nd Ser., **21**, 127-180 (KUGC IV).

Takase B., and Miyauchi-Isobe, N., 1986b, “Kiso Survey for Ultraviolet-Excess Galaxies V”, *Ann. Tokyo Astron. Obs*., 2nd Ser., **21**, 181-217 (KUGC V).

Takase B., and Miyauchi-Isobe, N., 1987a, “Kiso Survey for Ultraviolet-Excess Galaxies VI”, *Ann. Tokyo Astron. Obs*., 2nd Ser., **21**, 251-284 (KUGC VI).

Takase B., and Miyauchi-Isobe, N., 1987b, “Kiso Survey for Ultraviolet-Excess Galaxies VII”, *Ann. Tokyo Astron. Obs*., 2nd Ser., **21**, 363-386 (KUGC VII).

Takase B., and Miyauchi-Isobe, N., 1988, "Kiso Survey for Ultraviolet-Excess Galaxies VIII", *Ann. Tokyo Astron. Obs*., 2nd Ser., **22**, 41-58 (KUGC VIII).

Takase B., and Miyauchi-Isobe, N., 1989a, "Kiso Survey for Ultraviolet-Excess Galaxies IX", *Publ. Natl. Astron. Obs. Japan*, **1**, 11-42 (KUGC IX).

Takase B., and Miyauchi-Isobe, N., 1989b, "Kiso Survey for Ultraviolet-Excess Galaxies X", *Publ. Natl. Astron. Obs. Japan*, **1**, 97-125 (KUGC X).

Takase B., and Miyauchi-Isobe, N., 1990, "Kiso Survey for Ultraviolet-Excess Galaxies XI", *Publ. Natl. Astron. Obs. Japan*, **1**, 181-206 (KUGC XI).

Takase B., and Miyauchi-Isobe, N., 1991a, "Kiso Survey for Ultraviolet-Excess Galaxies XII", *Publ. Natl. Astron. Obs. Japan*, **2**, 7-36 (KUGC XII).

Takase B., and Miyauchi-Isobe, N., 1991b, "Kiso Survey for Ultraviolet-Excess Galaxies XIII", *Publ. Natl. Astron. Obs. Japan*, **2**, 37-61 (KUGC XIII).

Takase B., and Miyauchi-Isobe, N., 1991c, "Kiso Survey for Ultraviolet-Excess Galaxies XIV", *Publ. Natl. Astron. Obs. Japan*, **2**, 239-265 (KUGC XIV).

Takase B., and Miyauchi-Isobe, N., 1992a, "Kiso Survey for Ultraviolet-Excess Galaxies XV", *Publ. Natl. Astron. Obs. Japan*, **2**, 399-429 (KUGC XV).

Takase B., and Miyauchi-Isobe, N., 1992b, "Kiso Survey for Ultraviolet-Excess Galaxies XVI", *Publ. Natl. Astron. Obs. Japan*, **2**, 573-600 (KUGC XVI).

Takase B., and Miyauchi-Isobe, N., 1993a, "Kiso Survey for Ultraviolet-Excess Galaxies XVII", *Publ. Natl. Astron. Obs. Japan*, **3**, 21-43 (KUGC XVII).

Takase B., and Miyauchi-Isobe, N., 1993b, "Kiso Survey for Ultraviolet-Excess Galaxies XVIII", *Publ. Natl. Astron. Obs. Japan*, **3**, 169-257 (KUGC XVIII).

Takase, B., Noguchi, T., and Maehara H., 1983, "A Morphological Study of Ultraviolet-Excess Galaxies", *Ann. Tokyo Astron. Obs., 2nd Ser*., **19**, 440-462.

Tomita A., Takeuchi, T., Usui, T., and Saito, M., 1997, "Characteristics of Kiso Ultraviolet-Excess Galaxies", *Astron. J*., **114**, 1758-1770.

Zamorano, J., Rego, M., Gallego, J., Vitores, A. G., Gonzalez-Riestra, R., and Rodriguez-Caderot, G. 1994, "Study of Emission-Line Galaxies: Universidad Complutense Madrid List", *Astrophys. J. Suppl*., **95**, 387

Table V-2a. List of KUGs (A0465)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0309+309 | 3 9 12.8 | 30 54 11 | C: | 0.2 X 0.1 | 17.0: | L | |
| 2 | 0310+309 | 3 10 43.3 | 30 58 14 | Sp: | 0.3 X 0.2 | 17.0: | L | |
| 3 | 0313+329 | 3 13 18.7 | 32 57 19 | Sp: | 0.3 X 0.2 | 16.7: | L | |
| 4 | 0331+285 | 3 31 35.8 | 28 33 22 | C: | 0.3 X 0.2 | 15.7: | M | 5ZW354 |

Table V-2b. List of KUGs (A0755)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0354+083 | 3 54 16.2 | 8 22 16 | Sk | 0.8 X 0.7 | 14.9 | L | Z418.002,M+1-11-1 |

**Notes on individual galaxies given in Table V-2b (A0755)**

0354+083 : Many blue knots are located on the ring surrounding the red nuclear region.

Table V-2c. List of KUGs (A0827)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0356+066 | 3 56 41.1 | 6 37 28 | Sp | 0.3 X 0.2 | 15.5: | L | |
| 2 | 0356+065 | 3 56 50.7 | 6 28 17 | Sp | 0.4 X 0.2 | 16.0: | L | |
| 3 | 0402+042 | 4 2 52.2 | 4 16 36 | Sk: | 0.7 X 0.7 | 15.4 | L | Z418.009,M+1-11-10 |

**Notes on individual galaxies given in Table V-2c (A0827)**

0402+042 : A blue thin ring surrounds the nuclear region concentrically.

Table V-2d. List of KUGs (A1109)

Table V-2d. *List of KUGs (A1109)*

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0149-168 | 1 49 35.3 | -16 53 39 | Pi | 0.7 X 0.1 | 16.3: | M | |
| 2 | 0150-167A | 1 50 5.6 | -16 47 37 | Sp: | 0.3 X 0.1 | 16.2: | L | |
| 3 | 0150-167B | 1 50 11.0 | -16 45 48 | Sk | 0.7 X 0.4 | 14.0 | L | N725,M-3-5-25 |
| 4 | 0150-138 | 1 50 9.7 | -13 49 26 | Sk: | 0.6 X 0.3 | 15.3: | L | |
| 5 | 0150-147 | 1 50 51.2 | -14 45 16 | C | 0.2 X 0.2 | 17.0: | L | |
| 6 | 0151-168 | 1 51 27.6 | -16 48 43 | C | 0.3 X 0.2 | 15.5: | L | |
| 7 | 0151-140 | 1 51 30.3 | -14 4 56 | Sp | 0.3 X 0.2 | 15.5: | L | |
| 8 | 0152-123 | 1 52 14.0 | -12 19 53 | Sp: | 0.3 X 0.2 | 16.2: | L | |
| 9 | 0155-152 | 1 55 5.3 | -15 15 8 | ? | 0.6 X 0.3 | 15.0 | L | M-3-6-3 |
| 10 | 0156-146 | 1 56 41.7 | -14 40 54 | C: | 0.3 X 0.2 | 16.0: | L | |
| 11 | 0157-152 | 1 57 41.5 | -15 12 44 | Sp: | 0.3 X 0.2 | 16.3: | L | |
| 12 | 0157-128 | 1 57 41.6 | -12 49 9 | C: | 0.2 X 0.1 | 16.5: | L | |
| 13 | 0157-163 | 1 57 55.5 | -16 21 50 | Sp | 0.6 X 0.3 | 14.7: | L | |
| 14 | 0158-176 | 1 58 56.6 | -17 36 42 | C | 0.2 X 0.1 | 16.5: | L | |
| 15 | 0159-141 | 1 59 8.7 | -14 8 57 | Sp: | 0.3 X 0.2 | 16.0: | L | |
| 16 | 0159-139 | 1 59 31.1 | -13 58 25 | Sp: | 0.2 X 0.2 | 16.5: | L | |
| 17 | 0200-138 | 2 0 53.0 | -13 51 34 | C: | 0.2 X 0.2 | 16.7: | L | |
| 18 | 0201-128 | 2 1 4.8 | -12 50 49 | Sk | 0.4 X 0.3 | 15.5: | L | |
| 19 | 0202-126 | 2 2 55.4 | -12 38 3 | Sp | 0.3 X 0.2 | 16.2: | L | |
| 20 | 0203-152 | 2 3 18.1 | -15 16 53 | Sp: | 0.3 X 0.2 | 16.0: | L | |
| 21 | 0203-177 | 2 3 18.1 | -17 47 12 | C | 0.2 X 0.2 | 16.5: | L | |
| 22 | 0204-135 | 2 4 30.1 | -13 33 23 | Sp: | 0.2 X 0.2 | 16.8: | L | |
| 23 | 0207-132 | 2 7 18.2 | -13 12 14 | Sp: | 0.4 X 0.3 | 15.5: | L | |
| 24 | 0207-156 | 2 7 55.1 | -15 41 11 | Ig: | 0.4 X 0.2 | 15.5: | M | |
| 25 | 0208-160 | 2 8 14.1 | -16 0 33 | Sp | 0.6 X 0.3 | 14.5 | L | M-3-6-10 |
| 26 | 0210-157 | 2 10 20.5 | -15 44 20 | Sp | 0.3 X 0.3 | 15.3: | L | |
| 27 | 0210-159 | 2 10 51.9 | -15 55 32 | Sp | 0.3 X 0.2 | 15.5: | M | |
| 28 | 0212-134 | 2 12 5.2 | -13 29 44 | Sp: | 0.4 X 0.3 | 15.0: | L | M-2-6-42 |

**Notes on individual galaxies given in Table V-2d (A1109)**

0149-168 : A triple galaxy system(?) with central and western blue components.

0155-152 : Featureless.

0159-141 : Bright red central knot.

0212-134 : Bright central region.

Table V-2e. List of KUGs (A1177)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0031-217A | 0 31 43.7 | -21 42 51 | Sk | 1.3 X 1.0 | 13.5 | L | M-4-2-18 |
| 2 | 0031-217B | 0 31 46.7 | -21 45 9 | C | 0.2 X 0.2 | 15.0 | L | M-4-2-19 |
| 3 | 0033-199 | 0 33 9.2 | -19 58 30 | C | 0.2 X 0.2 | 16.5: | L | |
| 4 | 0034-200 | 0 34 29.2 | -20 1 48 | C: | 0.2 X 0.2 | 16.5: | L | |
| 5 | 0035-182 | 0 35 52.2 | -18 13 39 | Sp: | 0.4 X 0.2 | 15.5: | L | |
| 6 | 0037-190 | 0 37 8.8 | -19 5 27 | Sp: | 0.4 X 0.2 | 16.0 | L | M-3-2-33 |
| 7 | 0037-203 | 0 37 35.8 | -20 20 18 | Sk | 0.6 X 0.6 | 14.5: | L | |
| 8 | 0037-173 | 0 37 56.2 | -17 22 37 | Sk: | 0.3 X 0.3 | 16.5: | L | |
| 9 | 0038-184 | 0 38 20.7 | -18 28 10 | Sp | 0.4 X 0.2 | 16.0: | L | |
| 10 | 0038-189 | 0 38 30.6 | -18 58 20 | Sp: | 0.3 X 0.2 | 16.5: | L | |
| 11 | 0038-214 | 0 38 42.5 | -21 24 21 | C | 0.4 X 0.3 | 15.0: | L | |
| 12 | 0038-213 | 0 38 57.9 | -21 19 11 | Sp: | 1.1 X 0.3 | 13.0 | M | N216,M-4-2-35 |
| 13 | 0039-171A | 0 39 11.0 | -17 7 31 | C | 0.3 X 0.2 | 15.5 | L | M-3-2-39 |
| 14 | 0039-171B | 0 39 15.1 | -17 8 3 | Sk | 1.1 X 1.1 | 14.0 | L | M-3-2-40 |
| 15 | 0039-199 | 0 39 31.6 | -19 56 24 | C | 0.2 X 0.1 | 17.0: | L | |
| 16 | 0041-180 | 0 41 19.2 | -18 0 47 | Sp: | 0.8 X 0.2 | 15.0 | L | M-3-3-1 |
| 17 | 0042-192 | 0 42 1.7 | -19 16 59 | C: | 0.3 X 0.2 | 17.0: | L | |
| 18 | 0042-181 | 0 42 31.4 | -18 7 34 | C: | 0.2 X 0.2 | 17.0: | L | |
| 19 | 0042-209 | 0 42 47.2 | -20 59 23 | Sk: | 0.3 X 0.2 | 16.5: | L | |
| 20 | 0042-185 | 0 42 55.0 | -18 32 47 | C: | 0.2 X 0.2 | 16.5: | L | |
| 21 | 0043-191 | 0 43 13.1 | -19 6 42 | Sp | 0.4 X 0.2 | 16.0: | L | |
| 22 | 0043-208 | 0 43 28.8 | -20 52 57 | ? | 0.7 X 0.4 | 15.5: | L | |
| 23 | 0044-210 | 0 44 39.7 | -21 1 59 | Sk | 12.5 X 5.0 | 10.0 | L | N247,M-4-3-5 |
| 24 | 0044-219 | 0 44 45.2 | -21 58 54 | Sp: | 0.3 X 0.2 | 16.5: | L | |
| 25 | 0045-184 | 0 45 6.4 | -18 27 41 | C | 0.2 X 0.1 | 17.0: | M | |
| 26 | 0045-207A | 0 45 6.5 | -20 42 4 | Sk | 1.0 X 0.3 | 14.0 | M | M-4-3-10 |
| 27 | 0045-207B | 0 45 8.4 | -20 45 31 | Sk | 0.8 X 0.4 | 16.0 | L | M-4-3-11 |
| 28 | 0045-207C | 0 45 9.2 | -20 47 32 | Sk | 0.9 X 0.6 | 14.5 | M | M-4-3-13 |
| 29 | 0045-207D | 0 45 9.4 | -20 43 28 | Pi: | 0.4 X 0.2 | 17.0 | M | M-4-3-12 |
| 30 | 0045-182 | 0 45 10.0 | -18 16 3 | Sk: | 0.4 X 0.3 | 16.5: | L | |
| 31 | 0045-217 | 0 45 12.9 | -21 45 50 | Sp | 0.4 X 0.4 | 15.0: | M | |
| 32 | 0046-185 | 0 46 37.9 | -18 33 33 | C: | 0.2 X 0.2 | 16.5: | L | |
| 33 | 0046-207 | 0 46 55.3 | -20 42 13 | Sk: | 0.4 X 0.3 | 15.5: | L | |
| 34 | 0048-201 | 0 48 8.4 | -20 9 58 | Sp: | 0.2 X 0.2 | 16.5: | L | |
| 35 | 0048-209 | 0 48 11.6 | -20 55 22 | Sp: | 0.4 X 0.2 | 16.2: | M | |
| 36 | 0049-200 | 0 49 30.3 | -20 4 41 | C: | 0.1 X 0.1 | 17.5: | L | |
| 37 | 0049-210 | 0 49 36.0 | -21 1 19 | Sk: | 0.6 X 0.1 | 16.0: | L | |
| 38 | 0052-175 | 0 52 4.1 | -17 33 41 | Sk | 0.6 X 0.2 | 16.0: | L | |

**Notes on individual galaxies given in Table V-2e (A1177)**

0031-217A: Many blue clumps along the two thick arms.

0037-173 : A star is at the south edge of the galaxy.

0038-213 : The elongated central region possesses high surface brightness in U band.

0039-171A: Possibly pair with KUG 0039-171B.

0043-208 : The object is composed of a few fragments. Two stars are overlapped with it.

0044-210 : A number of blue knots are located on the galaxy plane.

0045-207A: Bright thick arms.

0045-207B: Barred spiral with two patchy arms.

0045-207C: Blue disk and arms.

0045A-207A,0045-207B,0045-207C,0045-207D : Possibly forms an interacting galaxy group.

0045-217 : Ring-like structure in the outer region.

0048-201 : A bright star on the edge of the galaxy.

Table V-2f. List of KUGs (A1178)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1* | 0048-209 | 0 48 11.7 | -20 55 22 | Sp: | 0.4 X 0.2 | 16.2: | M | |
| 2 | 0051-211 | 0 51 26.9 | -21 10 10 | Sp: | 0.3 X 0.1 | 16.5: | L | |
| 3 | 0053-173 | 0 53 51.0 | -17 18 56 | Sp | 0.6 X 0.1 | 16.0: | L | |
| 4 | 0055-186 | 0 55 34.6 | -18 39 15 | Ig: | 0.5 X 0.2 | 15.3: | M | |
| 5 | 0056-211 | 0 56 19.2 | -21 6 37 | Sk | 0.6 X 0.3 | 15.0 | L | N320,M-4-3-37 |
| 6 | 0056-186 | 0 56 39.1 | -18 38 49 | Sp: | 0.2 X 0.1 | 16.5: | L | |
| 7 | 0101-208 | 1 1 24.6 | -20 52 30 | Ic: | 0.7 X 0.7 | 14.0: | M | |
| 8 | 0102-179A | 1 2 40.5 | -17 59 41 | Sp: | 0.4 X 0.2 | 15.5 | L | M-3-3-21 |
| 9 | 0102-179B | 1 2 57.1 | -17 58 56 | Sk: | 0.5 X 0.3 | 15.2: | L | |
| 10 | 0103-200 | 1 3 10.3 | -20 3 40 | C | 0.2 X 0.2 | 16.5: | L | |
| 11 | 0105-178 | 1 5 8.9 | -17 48 20 | Sk: | 0.4 X 0.4 | 14.5 | L | M-3-4-1 |
| 12 | 0105-177 | 1 5 19.0 | -17 46 26 | Pi | 0.8 X 0.4 | 14.0 | M | I1622/I1623,V114A,V114B,M-3-4-3/M-3-4-4 |

**Notes on individual galaxies given in Table V-2f (A1178)**

0056-211 : Two elongated clumps are located in the north and south outer region.

0101-208 : Bright clumps are located on the galaxy.

0105-177 : A red central knot.

Table V-2g. List of KUGs (A1179)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0108-224A | 1 8 39.7 | -22 28 47 | Sp: | 0.3 X 0.1 | 16.5: | L | |
| 2 | 0108-224B | 1 8 40.5 | -22 26 16 | ? | 0.2 X 0.2 | 17.5: | L | |
| 3 | 0111-207 | 1 11 42.2 | -20 44 6 | Sp: | 0.3 X 0.1 | 16.5: | L | |
| 4 | 0112-171 | 1 12 10.2 | -17 8 33 | Sp: | 0.2 X 0.2 | 17.0: | L | |
| 5 | 0113-213 | 1 13 14.5 | -21 20 31 | Sp: | 0.2 X 0.1 | 17.5: | L | |
| 6 | 0114-171 | 1 14 7.3 | -17 7 56 | ? | 0.2 X 0.2 | 16.5: | L | |
| 7 | 0114-170 | 1 14 15.9 | -17 3 24 | Sp: | 0.4 X 0.2 | 16.0: | L | |
| 8 | 0116-173 | 1 16 35.4 | -17 19 21 | ? | 1.1 X 0.4 | 14.0 | L | I93,M-3-4-43 |
| 9 | 0117-184 | 1 17 19.7 | -18 25 12 | C | 0.1 X 0.1 | 17.0 | L | |
| 10 | 0117-226 | 1 17 44.8 | -22 38 30 | Ic: | 0.7 X 0.4 | 15.0 | M | N478,M-4-4-5 |
| 11 | 0118-176 | 1 18 8.8 | -17 39 26 | Pi | 1.2 X 0.4 | 14.5 | M | M-3-4-52/M-3-4-53 |
| 12 | 0118-194 | 1 18 15.8 | -19 27 1 | C | 0.2 X 0.1 | 17.5: | L | |
| 13 | 0119-176 | 1 19 20.9 | -17 41 6 | Pd | 0.2 X 0.2 | 16.0: | L | |
| 14 | 0120-177 | 1 20 44.6 | -17 47 3 | Sp: | 0.3 X 0.2 | 15.5 | L | M-3-4-58 |
| 15 | 0123-184 | 1 23 16.6 | -18 28 13 | Sp | 0.6 X 0.4 | 15.0 | L | M-3-4-64 |
| 16 | 0124-190 | 1 24 22.4 | -19 5 28 | C: | 0.4 X 0.3 | 16.5: | L | |
| 17 | 0124-209 | 1 24 28.6 | -20 59 26 | C: | 0.3 X 0.2 | 16.5: | L | |
| 18 | 0124-196 | 1 24 54.4 | -19 37 26 | C: | 0.1 X 0.1 | 17.0: | L | |
| 19 | 0127-209 | 1 27 9.6 | -20 57 42 | Ig: | 0.3 X 0.2 | 16.2: | M | |
| 20 | 0127-225A | 1 27 14.6 | -22 33 9 | Sp | 0.3 X 0.2 | 16.0 | L | M-4-4-18 |

**Notes on individual galaxies given in Table V-2g (A1179)**

0116-173 : An elongated blue clump is located in the northern part of the galaxy.

0117-226 : Patchy west main body and east blue clumps.

0118-176 : Pair with a northern non-KUG.

0119-176 : Pair with a southern non-KUG.

Table V-2h. List of KUGs (A1180)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0132-199 | 1 32 5.8 | -19 54 20 | C | 0.3 X 0.2 | 15.5: | L | |
| 2 | 0132-181 | 1 32 28.7 | -18 6 59 | C: | 0.3 X 0.3 | 14.0: | L | |
| 3 | 0140-206 | 1 40 5.7 | -20 38 14 | Sp: | 0.6 X 0.2 | 15.5: | L | |
| 4 | 0140-184 | 1 40 18.0 | -18 28 47 | Sk | 1.2 X 0.6 | 14.0 | L | M-3-5-16 |
| 5 | 0150-191 | 1 50 52.0 | -19 11 4 | Sp | 1.2 X 0.2 | 14.5 | L | M-3-5-28 |

Table V-2i. List of KUGs (A1250)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0051-238 | 0 51 51.9 | -23 50 23 | C: | 0.2 X 0.2 | 16.5: | L | |
| 2 | 0054-236 | 0 54 33.7 | -23 36 58 | C: | 0.3 X 0.2 | 16.0: | L | |
| 3 | 0101-280 | 1 1 22.1 | -28 1 19 | Ic | 0.6 X 0.3 | 15.0: | L | |

Table V-2j. List of KUGs (A1251)

| No. | KUG-NAME | R. A. (1950.0) | DEC. | MOR. TYPE | APP. SIZE | APP. MAG. | UVX DEG. | OTHER NAME(S) |
|---|---|---|---|---|---|---|---|---|
| 1 | 0112-256 | 1 12 3.1 | -25 38 52 | Sp: | 1.0 X 0.4 | 15.0 | L | M-4-4-1 |
| 2 | 0112-268 | 1 12 29.7 | -26 50 31 | C: | 0.3 X 0.2 | 15.5: | L | |
| 3 | 0113-267 | 1 13 10.0 | -26 42 45 | Sp | 1.3 X 0.2 | 15.0 | M | M-5-4-24 |
| 4 | 0115-223 | 1 15 10.3 | -22 21 5 | C: | 0.2 X 0.1 | 17.0: | L | |
| 5 | 0116-242 | 1 16 21.6 | -24 12 28 | Ic: | 1.0 X 0.4 | 14.0 | M | M-4-4-3 |
| 6 | 0116-257 | 1 16 50.0 | -25 47 36 | Sp | 0.6 X 0.5 | 15.0: | L | |
| 7* | 0117-226 | 1 17 44.7 | -22 38 26 | Ic: | 0.7 X 0.4 | 15.0 | M | N478,M-4-4-5 |
| 8 | 0127-225A | 1 27 14.7 | -22 33 9 | Sp | 0.3 X 0.2 | 16.0 | L | M-4-4-18 |
| 9 | 0127-225B | 1 27 16.6 | -22 35 0 | Sp | 0.7 X 0.2 | 15.5 | L | M-4-4-19 |
| 10 | 0128-229A | 1 28 1.8 | -22 54 46 | Sp | 0.4 X 0.1 | 16.5: | L | |
| 11 | 0128-229B | 1 28 5.5 | -22 55 28 | Sk | 4.5 X 2.5 | 11.5 | L | N578,M-4-4-20 |
| 12 | 0129-258 | 1 29 18.4 | -25 48 12 | Sp: | 0.4 X 0.2 | 16.0: | L | |
| 13 | 0131-246 | 1 31 48.4 | -24 37 12 | C: | 0.3 X 0.2 | 16.5: | L | |

**Notes on individual galaxies given in Table V-2j (A1251)**

0112-256 : Blue bar/arm structure with the red central region.

0113-267 : A knot and the body of the galaxy is blue in the north-west region.

0116-242 : A very blue north-east clump.

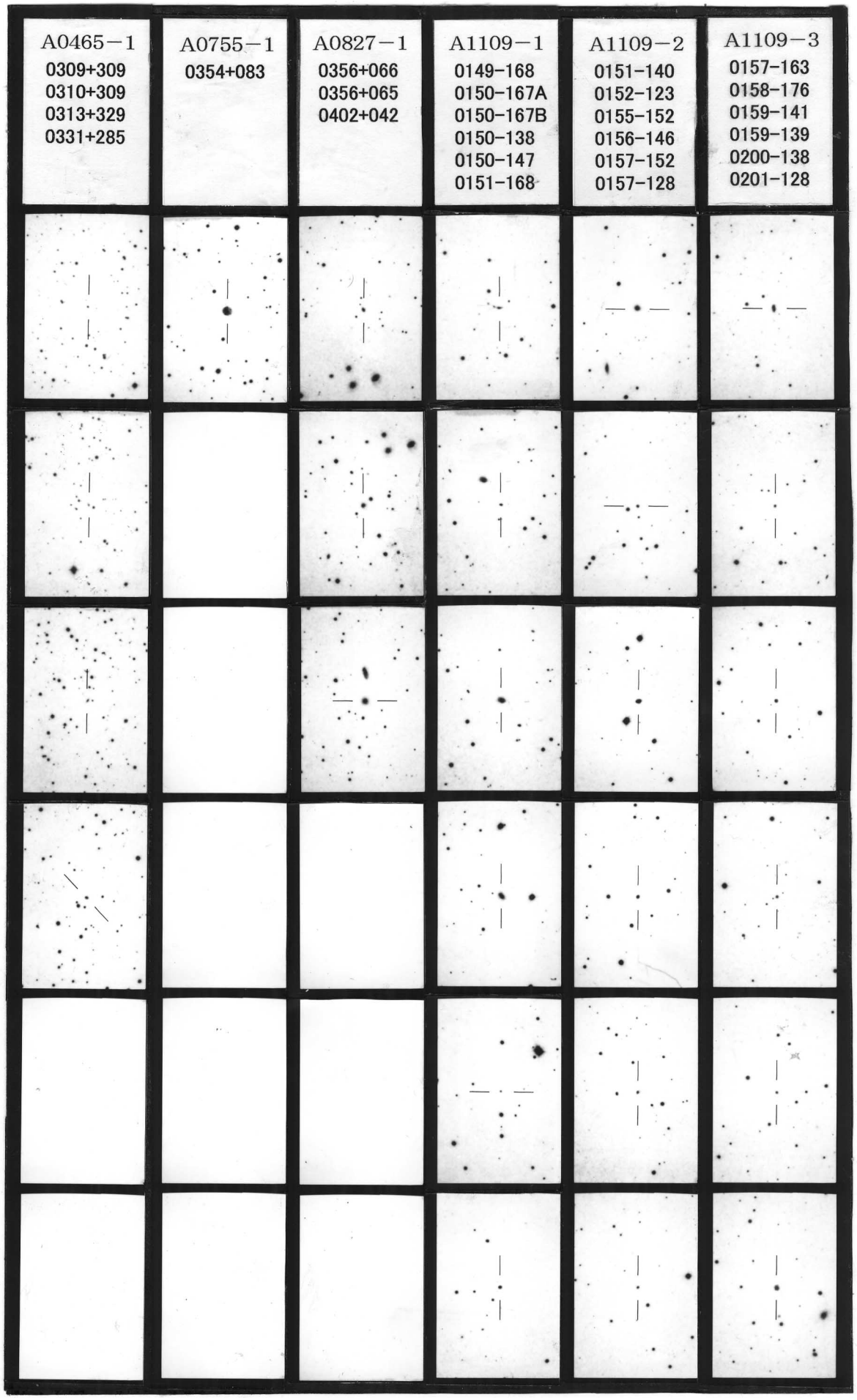
0116-257 : The central portion and a southern arm are blue.

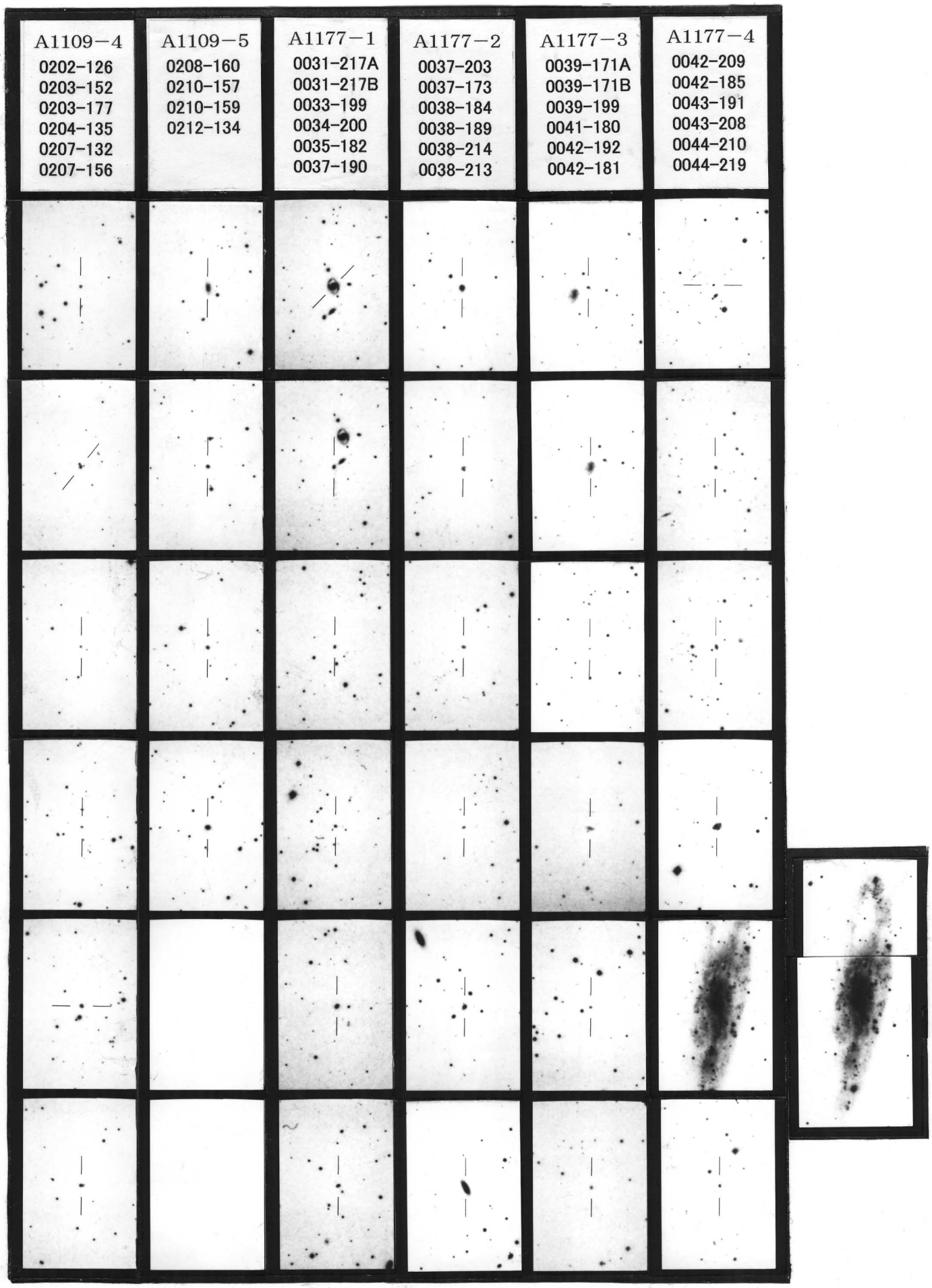
0117-226 : East clumps are blue, while the west bright portion is red.

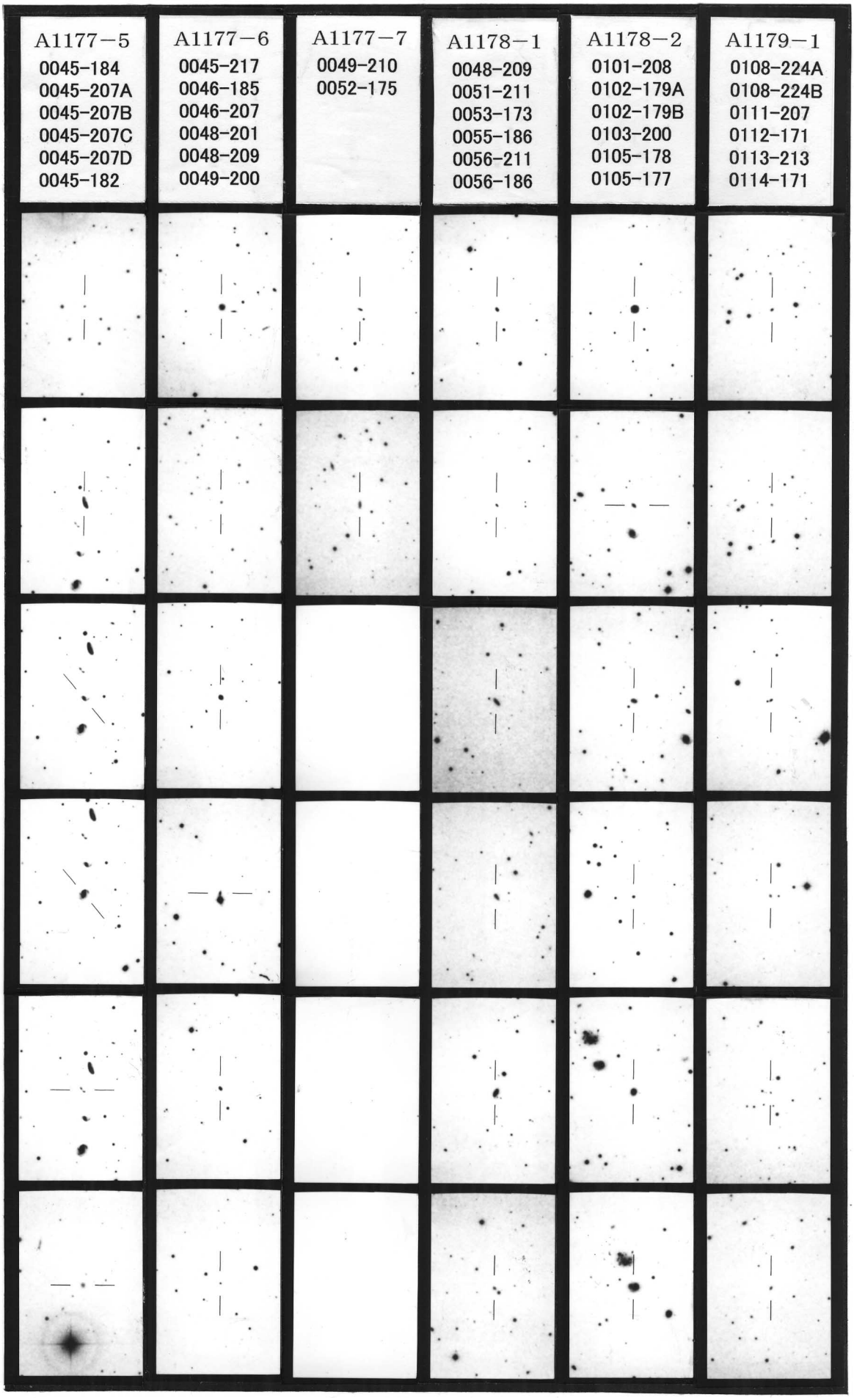
0128-229A: Overlapped with the outer disk of KUG 0128-229B.

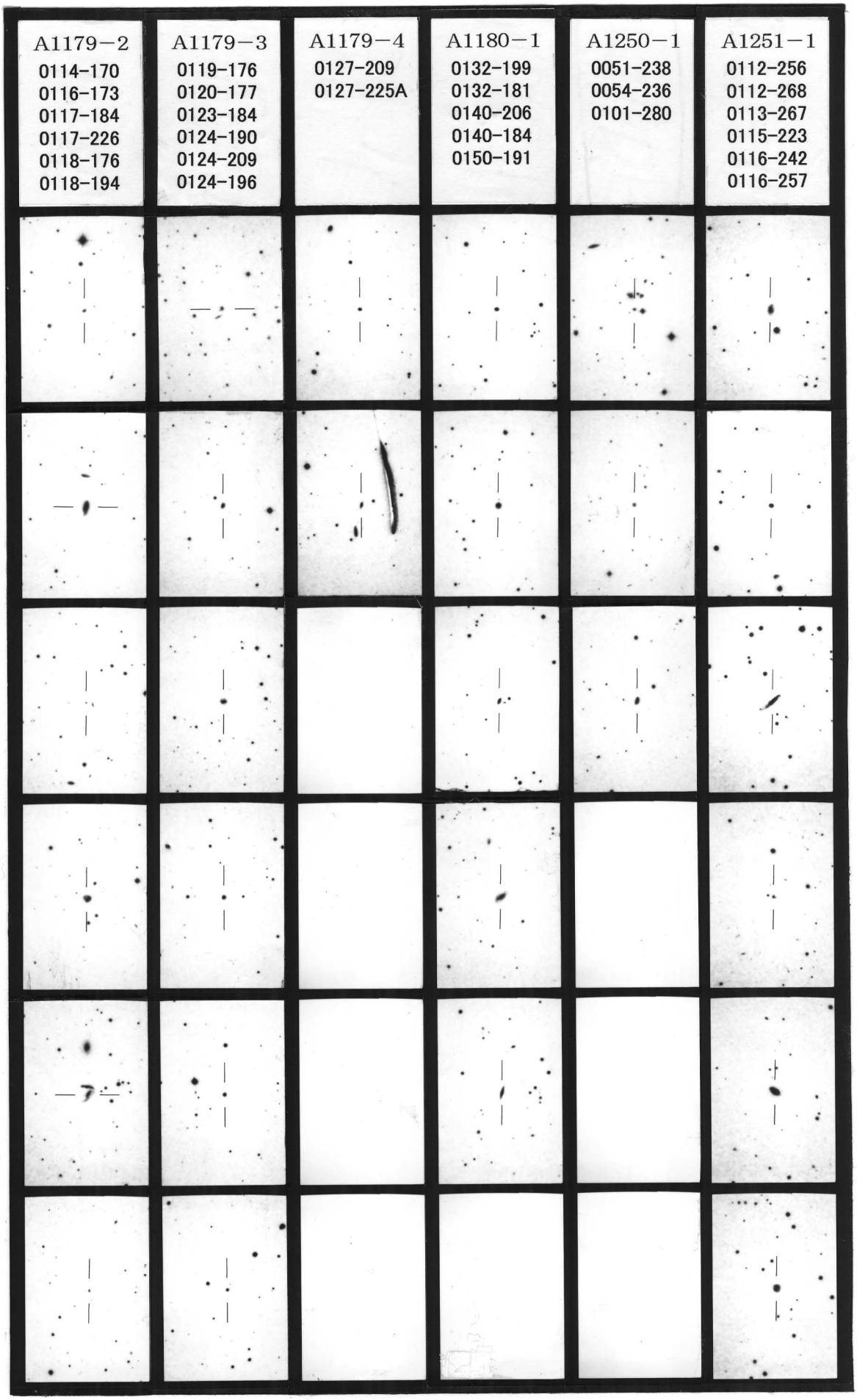
Figure V-1. Finding Charts

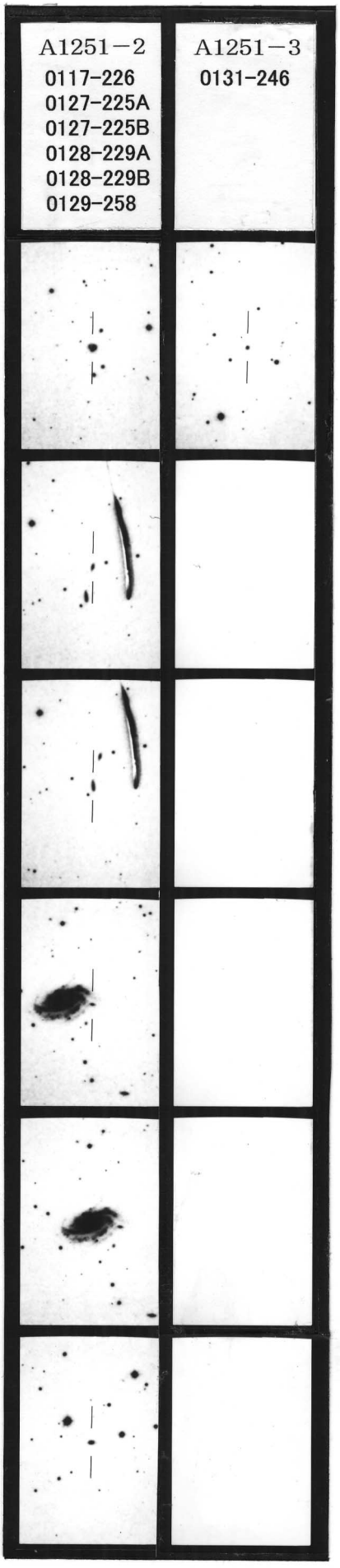
In the following pages, finding charts are shown for each KUG listed in the catalogue (Table V-2). These photographs are reproduced from the Palomar Sky Survey blue prints ( © 1960 National Geographic Society－Palomar Sky Survey reproduced by permission of the California Institute of Technology). The chart is in magnification of 3.0 times (0.34′/mm), and the field of 11.8′ ×7.7′. The north is up, east to the left.


A0465−1
0309+309
0310+309
0313+329
0331+285
A0755−1
0354+083
A0827−1
0356+066
0356+065
0402+042
A1109−1
0149−168
0150−167A
0150−167B
0150−138
0150−147
0151−168
A1109−2
0151−140
0152−123
0155−152
0156−146
0157−152
0157−128
A1109−3
0157−163
0158−176
0159−141
0159−139
0200−138
0201−128


A1109−4
0202−126
0203−152
0203−177
0204−135
0207−132
0207−156
A1109−5
0208−160
0210−157
0210−159
0212−134
A1177−1
0031−217A
0031−217B
0033−199
0034−200
0035−182
0037−190
A1177−2
0037−203
0037−173
0038−184
0038−189
0038−214
0038−213
A1177−3
0039−171A
0039−171B
0039−199
0041−180
0042−192
0042−181
A1177−4
0042−209
0042−185
0043−191
0043−208
0044−210
0044−219


A1177−5
0045−184
0045−207A
0045−207B
0045−207C
0045−207D
0045−182
A1177−6
0045−217
0046−185
0046−207
0048−201
0048−209
0049−200
A1177−7
0049−210
0052−175
A1178−1
0048−209
0051−211
0053−173
0055−186
0056−211
0056−186
A1178−2
0101−208
0102−179A
0102−179B
0103−200
0105−178
0105−177
A1179−1
0108−224A
0108−224B
0111−207
0112−171
0113−213
0114−171


A1179−2
0114−170
0116−173
0117−184
0117−226
0118−176
0118−194
A1179−3
0119−176
0120−177
0123−184
0124−190
0124−209
0124−196
A1179−4
0127−209
0127−225A
A1180−1
0132−199
0132−181
0140−206
0140−184
0150−191
A1250−1
0051−238
0054−236
0101−280
A1251−1
0112−256
0112−268
0113−267
0115−223
0116−242
0116−257



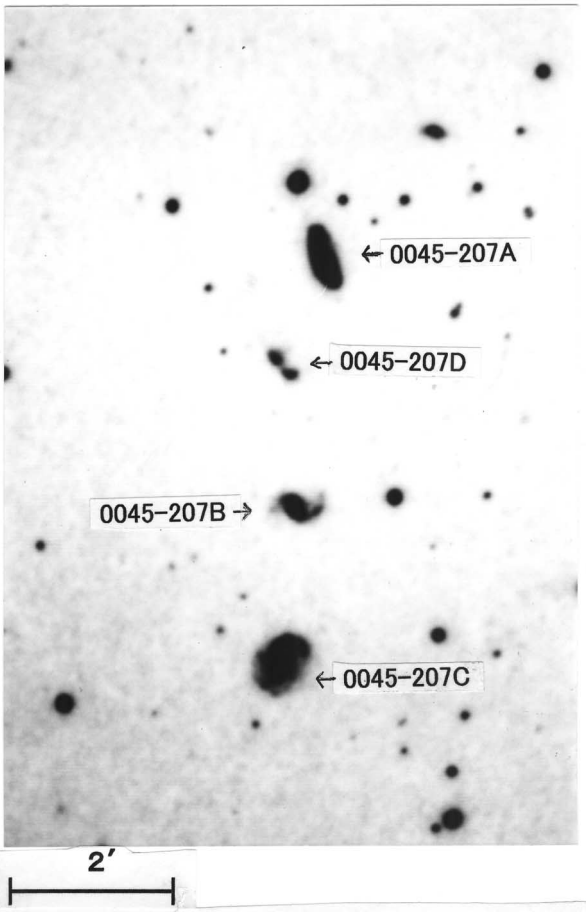




Figure V-2. KUG group of galaxies in SCG0045-2043

Publ. Natl. Astron. Obs. Japan Vol. 8. 17–31 (2005)

# Inaccuracies of Trigonometric Functions in Computer Mathematical Libraries

Takashi ITO and Sadamu KOJIMA*

(Received April 28, 2005)

**Abstract**

Recent progress in the development of high speed computers has enabled us to perform larger and faster numerical experiments in astronomy. However, sometimes the high speed of numerical computation is achieved at the cost of accuracy. In this paper we show an example of accuracy loss by some mathematical functions on certain computer platforms in Astronomical Data Analysis Center, National Astronomical Observatory of Japan. We focus in particular on the numerical inaccuracy in sine and cosine functions, demonstrating how accuracy deterioration emerges. We also describe the measures that we have so far taken against these numerical inaccuracies. In general, computer vendors are not eager to improve the numerical accuracy in the mathematical libraries that they are supposed to be responsible for. Therefore scientists have to be aware of the existence of numerical inaccuracies, and protect their computational results from contamination by the potential errors that many computer platforms inherently contain.

Key words: numerical simulation, trigonometric function, celestial mechanics

## 1. Introduction

Astronomy is a science of observation. Most astronomical phenomena evolve over extremely long periods, and our observations tend to give the impression that the universe is static. Because of this apparent stasis, and also because of the extremely large spatial scale of astronomical phenomena, astronomy has progressed as a mostly passive science, unlike physics and chemistry where scientists can perform fullscale real experiments. The only way astronomers can control the boundary and initial conditions of the phenomena in the universe is through numerical experiments using computers. Therefore, numerical experiments are far more important in astronomy than in other sciences.

Recent remarkable progress in digital technology has enabled us to perform huge and very fast numerical experiments on computers. The size of numerical experiments in astrophysics is getting larger, but these experiments are taking less time because of the increasing speed of computers. We often refer to a computer system used for numerical experiments as a "telescope for theoretical astronomy." The evolution rate of these telescopes is still going up, and now we are able to compare the output from these theoretical telescopes with that from real telescopes. In Astronomical Data Analysis Center (hereafter called ADAC) of National Astronomical Observatory of Japan (hereafter called NAOJ), we have been operating a large computer system for numerical astronomy since 1996 (Ito, 1997). Now a vector/parallel computer and a cluster of fast PCs together with the special-purpose computer for gravitational *N*-body problem, "GRAPE", serve as our theoretical telescope, which is being nearly fully used by many clients for their research. As for the details of the computer system in ADAC, visit our webpage at `http://www.cc.nao.ac.jp/` .

Although the progress in computer technology is indeed marvelous, there is one thing we should take note of: High speed computation is sometimes achieved at the cost of numerical accuracy. High accuracy generally requires cumbersome procedures: higher-order polynomial approximation, careful handling of round-off errors, use of huge and complicated numerical tables for evaluating function values, and so on. These treatments could cost a great deal of computation time, which would run counter to efforts to achieve high performance. The accuracy of floating-point structure is standardized by IEEE (the Institute of Electrical and Electronics Engineers, `http://www.ieee.org/`), especially by the so-called IEEE 754, and major computer products are supposed to adhere to this standardization. However, IEEE 754 does not specify the accuracy of transcendental mathematical functions such as `sin()`, `cos()`, `log()`, or `exp()` that are provided by each computer vendor; customers are obliged to believe what computer vendors say as to the accuracy of these functions. And, as we easily might anticipate, sometimes these mathematical functions do not possess sucient accuracy in terms of scientic research. Even if the numerical error in a single arithmetic operation is very small, it could pile up and lead to a devastating result, producing a totally dierent solution in a computer simulation.

In this paper we exemplify some inappropriate numerical results caused by numerical inaccuracy in trigonometric math libraries on certain computer platforms in ADAC/NAOJ, and describe how we have dealt with the problem so that clients' numerical research is performed without a serious flaw. At first in Section 2, we demonstrate numerical errors that are typical of those we found in a numerical integration involving trigonometric functions, showing how the accuracy loss took place. Then, the cause of the numerical inaccuracy is further dug up in Section 3. We found that slight differences in the result of numerical

*EXEC Corporation, Tsuji 1-9-1-214, Saitama 336-0025, Japan

trigonometric subroutines had ended up in a large and secular numerical error in some numerical integrations. In Section 4, we explain the measures that we have taken against the numerical errors. Porting a dependable mathematical library such as GNU libm is always a good solution. Section 5 is devoted to giving summary and discussion on the whole subject. All through this manuscript, we use typewriter face for programming related (or source code related) items such as function names (ex. `sin()`), filenames (ex. `glib math.h`), or command names (ex. `gcc -O4`).

According to our experience, computer vendors tend to pay a great deal of attention to the high performance and capacity of their products, but not to the accuracy of basic mathematical libraries which play a significant role in scientic and engineering calculation. Hence scientists have to protect their computational results from contamination by the potential numerical inaccuracy that many platforms could inherently contain: we are totally on our own as to whether or not our numerical result is accurate enough with respect to our purpose. This manuscript represents our first and preliminary report dealing only with a small part of the potentially huge problem of built-in numerical inaccuracies of mathematical subroutines on various platforms we operate in ADAC. We are going to continue examining the numerical inaccuracies in our computer system and improving the measures we use to prevent these inaccuracies, details of which we will publish as future reports.

## 2. Example of numerical error

In July 1998, ADAC upgraded the operating system of the Sun SPARC workstations that were openly used for our clients. This incident made us aware of the existence of numerical inaccuracy in some of the built-in trigonometric functions. In this section we briefly describe an example of the accuracy loss we experienced after the upgrade.

Soon after we upgraded the operating system from Solaris 2.5.1 to Solaris 2.6, we noticed that some of the numerical results that we obtained differed from those obtained before. We performed many kinds of comparison tests, and found that in the new system, the accuracy of a certain sort of numerical integration was definitely lower than it used to be. More specifically, what our numerical test told us was: the total angular momentum of gravitational *N*-body systems was no longer conserved in a certain type of numerical integration. It had been very well preserved until the operating system was upgraded in July 1998. The non-conservation of total angular momentum is illustrated in Fig. 1, which we will soon describe in detail.

The flawed calculation is just a regular numerical integration of a set of ordinary dierential equations that describes the dynamics of a gravitational *N*-body system. The purpose of the integration is to track the orbital motion of the solar system planets over a long timespan along the path described by Newton's equations of motion. These kinds of calculations are very common, seen almost everywhere in the astronomical field.

In this particular integration, a symplectic integrator is used. Since many of the systems that celestial mechanics deals with compose a Hamiltonian system or its proximity, numerical integration schemes specifically designed to maintain the Hamiltonian structure are not only desirable but promising. Symplectic integrators are exactly the method that satisfies such a requirement. In general, symplectic integrators conserve the total energy of the system quite well, preventing artificial dumping or excitation due to an accumulation of local truncation errors. Since the 1990s when they were introduced to the field of astronomy and astrophysics, symplectic integrators have definitely been one of the most significant numerical integration schemes in solar system dynamics.

Major symplectic integrators have a very important property in addition to the preservation of total energy: Explicit symplectic integrator rigorously preserves the total angular momentum within the range of round-off errors (Yoshida, 1990). This fact comes from the characteristic commutation of an integral $\Phi(q, p)$ of a Hamiltonian $H = T(p) + V(q)$ to both kinetic energy $T(p)$ and potential energy $V(q)$ where $(q, p)$ are the canonical coordinate and momentum, such as

$$\{\Phi(q, p), T(p)\} = \{\Phi(q, p), V(q)\} = 0 \quad (1)$$

Though the energy integration does not have this property, the momentum integrals and the angular momentum integrals of gravitational *N*-body systems do. It is obvious that this class of integrals is exactly conserved by the explicit symplectic integrators. This characteristic is supposed to guarantee that the symplectic numerical error of the total angular momentum in a gravitational *N*-body system has no truncation error. Therefore conservation of the total angular momentum in a gravitational *N*-body system can be a very good testbed for numerical accuracy.

As a standard accuracy test, we took a dynamical model of our solar system consisting of nine planets with the current orbital elements (Standish, 1990). We integrated the orbits of all planets for about 80,000 years using the Wisdom–Holman symplectic map (Wisdom and Holman, 1991; Kinoshita *et al.*, 1991). The source code is written in C. This calculation is almost the same as what Ito & Tanikawa (2002) did as a series of very long-term numerical integrations of solar system planetary orbits over more than 4 billion years. A quick comparison of panels (a) and (b) in Fig. 1 dramatically illustrates how the accuracy loss appeared after the operating system upgrade. Fig. 1 (b) shows the relative error of total angular momentum in this integration on the old platform, Solaris 2.5.1. We can see a good conservation of total angular momentum at the order of $10^{-13}$ (panel (b) in Fig. 1 is magnified as panel (b) in Fig. 2). On the other hand, Fig. 1 (a) shows the same calculation result computed on the new (current) platform, Solaris 2.8. Now it is clear that the total angular momentum is not conserved, creating a nearly linear secular error.

Here we have to mention a potential complication: The result in Fig. 1 (a) was actually not calculated on the Solaris 2.6 platform that we mentioned before, but on the newer Solaris 2.8 platform that we operate in 2005. But we have confirmed that any Solaris platform newer than 2.6 yields the same accuracy loss in this numerical integration as the

Solaris 2.6 platform does. Also, the version of C compiler is different before and after the operating system upgrade, as we will see later in Table 1. Hence, readers might wonder whether the accuracy loss was really caused by the upgrade, and not by the change in compiler version. Here also, we have confirmed that the accuracy loss occurring on this platform does not depend upon the compiler, whether we use Fujitsu C, SUNWorks Pro C, or GNU C. When you compile C source codes on these platforms, mathematical libraries are dynamically linked to runtime linkers for dynamic objects such as `/usr/lib/libm.so.1` which are supplied by the operating system. This means you would obtain almost the same result, i.e. almost the same accuracy loss in this integration, even if you use different compilers. (There could be a slight difference in numerical result depending on how each compiler optimizes its arithmetic operations.)

We then applied this accuracy test to the wide variety of computer platforms listed in Table 1. We tested a couple of compile options for optimization, but no significant difference appeared as to the numerical error of angular momentum conservation, whatever optimized option we used. In Fig. 1, obviously there are two groups of results that have produced different kinds of numerical errors in total angular momentum. The first one, including panels (a)(d)(g)(j)(k), produces linear-growing numerical errors. The second group, including panels (b)(c)(e)(f)(h)(i)(l), produces much smaller errors. What quickly draws our attention is that using the same hardware platform, newer operating systems tend to display worse results: For example, the newer Solaris (2.6 or later) yields a clear secular numerical error as you can see in Fig. 1(a), while the result from the older Solaris (2.5.1) looks much better (Fig. 1(b)). We see the same trend as to SGI IRIX whose older version (IRIX 6.2; Fig. 1(e)) had produced a much better result than the newer version (IRIX 6.5; Fig. 1(d)). An exception came from Fujitsu's vector/parallel computer, the VPP series: the

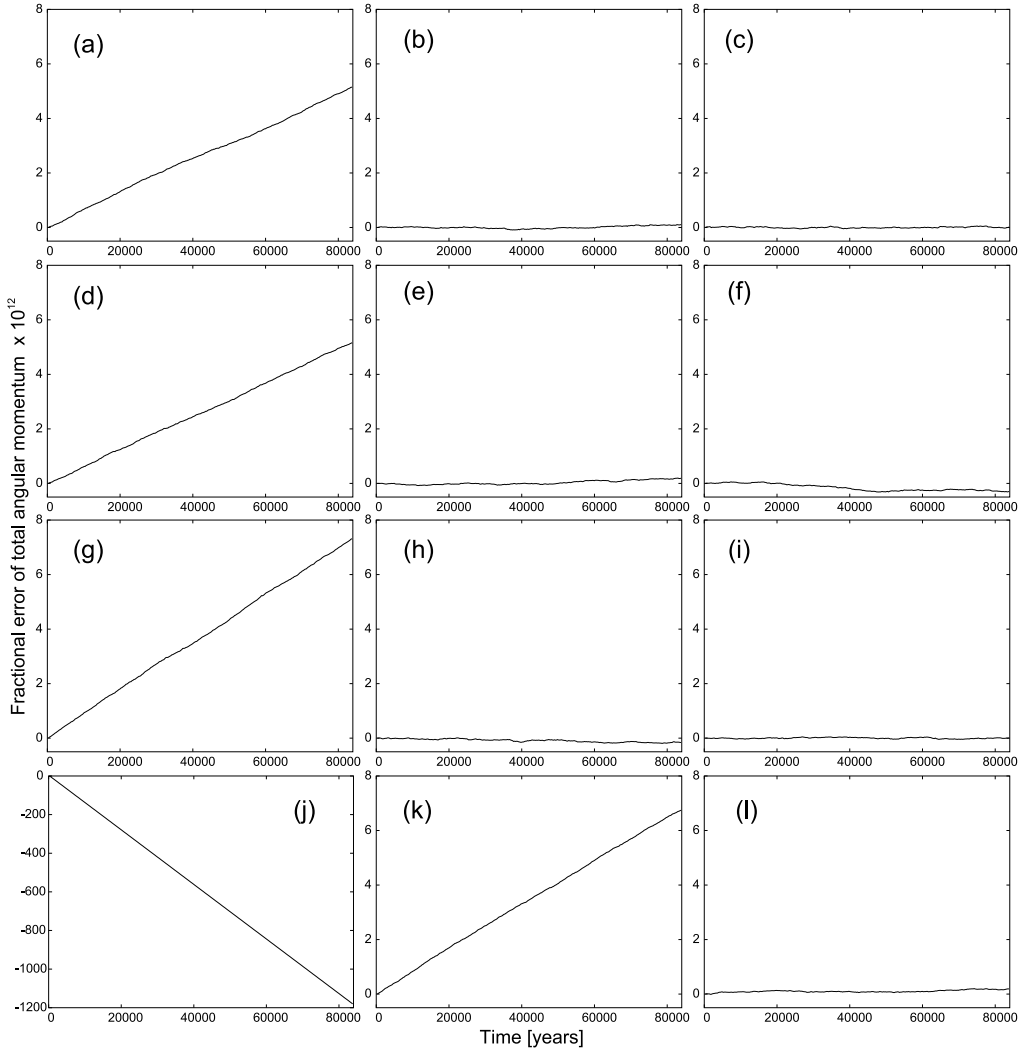



Fig. 1. Numerical errors of total angular momentum in a nine-planet system that are theoretically zero within the accuracy of round-off errors. The vertical scale is magnified $10^{12}$ times. (a) Solaris 2.8 + Fujitsu C 5.1.1, (b) Solaris 2.5.1 + Fujitsu C 4.0.2, (c) FreeBSD 5.3–RELEASE + GNU C 3.3.3, (d) IRIX 6.5 + MIPSpro C 7.3.1, (e) IRIX 6.2 + MIPSpro C 7.3, (f) RedHat 9 + Intel C 7.1, (g) RedHat 7.1 + Compaq C V6.4.9 on Alpha, (h) HP–UX 10.20B + HP C A.10.32.03, (i) HI–UX/MPP + Optimized C on Hitachi SR8000, (j) UXP/V V10L10 + C/VP V10L10 on Fujitsu VPP300, (k) Super–UX R9.1 Rev. 1 + C/SX V2 Rev. 4.01 on NEC SX–5, and (l) UXP/V V20L10 + C V20L20 on Fujitsu VPP5000. See Table 1 for more detail. Note that the panel (j) for UXP/V V10L10 with C/VP V10L10 on Fujitsu VPP300 has a much larger vertical scale ($\times 150$).

older version (VPP300; Fig. 1(j)) had produced extremely large (two orders of magnitude larger) numerical errors compared with any other platform. The newer version, VPP5000, seems much better (Fig. 1(l)) as far as this test is concerned.

If you take a closer look at the panels of the "good" group (b)(c)(e)(f)(h)(i)(l) that yield small numerical errors in Fig. 1, you can see slight differences in each of them (Fig. 2). Some of the errors might seem to be growing over a longer timescale, leading us to suppose that it might lead to a large secular numerical error sooner or later, as we see in the "bad" group of panels, Fig. 1 (a)(d)(g)(j)(k). But fortunately, previous research has demonstrated that most of the fluctuating errors shown in Fig. 2 will not secularly increase. For example, Ito & Tanikawa (2002) confirmed a very long-term dynamical stability of the solar system planets over more than 4 billion years using exactly the same numerical integration scheme with the same dynamical model as we used for the accuracy test described here. Ito & Tanikawa (2002) used several different computer platforms for their integrations: FreeBSD with GNU C, Solaris 2.5.1 with Fujitsu C, HP–UX, and RedHat with Intel C. As far as their long-term numerical integrations are concerned, the numerical error in total angular momentum of this system becomes no larger than the order of $10^{-11}$ showing no secular error over $\sim 10^{10}$ years. Fig. 3 is an example of the conservation of total angular momentum of the nine-planet system used in Ito & Tanikawa (2002) calculated on a Solaris 2.5.1 platform using Fujitsu C. The average error rate in their calculation is roughly $\sim 10^{-11}/10^{10}$ years = $10^{-21}$ /year, which is much smaller than that of $\sim 10^{-16}$ /year of the panels (a)(d)(g)(k) in Fig. 1, or $\sim 1.5 \times 10^{-14}$ /year of Fig. 1 (j).

### 3. Verification of the error cause

Browsing through the algorithm of the Wisdom–Holman symplectic map, we guessed that the numerical error described in the previous section perhaps was caused by numerical trigonometric functions, especially `sin()` and `cos()`. In the Wisdom–Holman symplectic map, the whole operation of transforming from Cartesian coordinates to orbital elements, advancing the mean anomaly of a particle, and transforming back is efficiently encapsulated in Gauss' famous $f$ and $g$ functions (Danby, 1992). In this procedure, a value $f\dot{g} - \dot{f}g$ is conserved as unity as

$$f\dot{g} - \dot{f}g = \cos^2\Delta u + \sin^2\Delta u = 1 \tag{2}$$

over every drift step through which each planet moves along its Keplerian orbit around central mass. Here $\Delta u$ is the difference in the eccentric anomaly of a planet over a stepsize. Suppose a celestial body that orbits around its central mass has the velocity $\boldsymbol{v}$ and position $\boldsymbol{r}$ at time $t$. The angular momentum $\boldsymbol{L}$ of the body per unit mass is

$$\boldsymbol{L} = \boldsymbol{r} \times \boldsymbol{v}. \tag{3}$$

$\boldsymbol{r}$ and $\boldsymbol{v}$ at time $t$ are expressed by $f$ and $g$ as (Danby, 1992)

$$\boldsymbol{r} = f\boldsymbol{r}_0 + g\boldsymbol{v}_0, \quad v = \dot{f}\boldsymbol{r}_0 + \dot{g}\boldsymbol{v}_0, \tag{4}$$

where $\boldsymbol{v}_0$ and $\boldsymbol{r}_0$ are the (constant) velocity and position vectors at the initial time, $t = 0$. So $L$ in (3) is expressed as

$$\begin{aligned}\boldsymbol{L} &= \boldsymbol{r} \times \boldsymbol{v} \\ &= (f\boldsymbol{r}_0 + g\boldsymbol{v}_0) \times \left(\dot{f}\boldsymbol{r}_0 + \dot{g}\boldsymbol{v}_0\right) \\ &= f\dot{f}(\boldsymbol{r}_0 \times \boldsymbol{r}_0) + f\dot{g}(\boldsymbol{r}_0 \times \boldsymbol{v}_0) + g\dot{f}(\boldsymbol{v}_0 \times \boldsymbol{r}_0) + g\dot{g}(\boldsymbol{v}_0 \times \boldsymbol{v}_0) \\ &= \left(f\dot{g} - \dot{f}g\right)(\boldsymbol{r}_0 \times \boldsymbol{v}_0).\end{aligned} \tag{5}$$

If there is no numerical error and (2) holds, the last equation in (5) becomes

$$\boldsymbol{r} \times \boldsymbol{v} = \boldsymbol{r}_0 \times \boldsymbol{r}_0, \tag{6}$$

which exactly denotes the conservation of angular momentum. On the other hand, if (2) does not hold by numerical error and $f\dot{g} - \dot{f}g$ gets less than or more than unity, the relationship (6) is not satisfied and angular momentum is no longer conserved. Based on this consideration, what we did next was to check out how close to unity the numerical value $\cos^2 x + \sin^2 x$ is on our computer platforms. In particular, we focused on the dependence of results on operating systems.

Due to the small eccentricities of solar system planetary orbits, it is usually safe for us to use a constant timestep (i.e. a nearly constant stepsize in eccentric anomaly, $u$) for the numerical integration of planetary orbits, which is ten days in this integration. This is nearly equivalent to $2\pi \times 10/365.25 = 0.172$ radian for Earth's orbit. However, the planet that possesses a significant amount of angular momentum and is largely responsible for the numerical error is Jupiter, whose orbital period is about 11.862 years. Then the 10-day stepsize is nearly equal to $2\pi \times 10/(365.25\ 11.862) = 0.0145$ ($\sim \pi/217$) radian for Jupiter's orbit. So we calculated the value $\cos^2 x + \sin^2 x - 1$ in the $x$ range of $[-\pi/128, \pi/128]$ using the stepsize of $2^{-27} \sim 7.450580596923828125 \times 10^{-9}$ radian with double-precision arithmetic.

The results of this test are summarized in Fig. 4. In Fig. 4, deviation from zero of $\cos^2 x + \sin^2 x - 1$ (which theoretically must be exactly zero) is normalized by the machine epsilon of double-precision arithmetic, $\varepsilon \sim 1.110223024625\ 1565 \times 10^{-16}$. We counted the relative frequency of each error value described in Fig. 4, making histograms as in Fig. 5. Machine epsilon, as well as ulp (unit in last place), represents the most basic unit when we deal with numerical errors in floating-point operations. We briefly summarized the concept of floating-point formats and the IEEE 754 standard in Appendix A.1 and A.2.

In the left three panels, (a)(b) and (c) of Fig. 4, the numerical deviation of $\cos^2 x + \sin^2 x - 1$ from zero is distributed almost symmetrically along the horizontal $y = 0$ line. This is also obvious from the left three symmetric histograms in Fig. 5. However if we look at the right three pan-

[1] Note that the sign of the average numerical errors shown in Fig. 4 might not necessarily have a direct relation to the sign of the secular numerical errors in Fig. 1, because we do not explicitly calculate $f\dot{g} - \dot{f}g = \cos^2\Delta u + \sin^2\Delta u$ in (2) when we compute the drift step in the Wisdom–Holman symplectic map. We are still investigating the detailed relationship between the sign of the numerical errors in Fig. 4 and that in Fig. 1.

els, (d)(e) and (f), we immediately notice the asymmetry of the numerical deviation along the $y = 0$ line, especially in panels (e) and (f).

The most typical asymmetry comes from IRIX 6.5 with MIPSpro C (Fig. 4(f)). On this platform, all the numerical errors on the positive side are +2 (in units of double-precision arithmetic machine epsilon) while the errors on the negative side are –1. This means that if you choose arguments $x$ of trigonometric functions `sin()` and `cos()` repeatedly and randomly from this range, $[-\pi/128,\pi/128]$, the average value of $\cos^2 x + \sin^2 x - 1$ would be positive, resulting in $\cos^2 x + \sin^2 x - 1 > 0$. This asymmetry is also clearly observed in Fig. 5 (f) where we see a high frequency bar at the horizontal position (i.e. error value) of +2. This is a possible cause of the secular error in angular momentum in the planetary nine-body problem mentioned in the previous section through equations (2) and (5)[1] .

The Solaris 2.8 platform running Fujitsu C (Fig. 4(e)) shows a similar pattern. In the range of $|x| \lesssim 0.01$, numerical errors are distributed almost symmetrically along $y = +1$, 0, –1 lines on this platform. But when $|x| \gtrsim 0.01$, we see the error along $y = +2$ as well. If you take several $x$ values around $x \sim 0.0145$ (which is close to the stepsize of eccentric anomaly of Jupiter's orbit, as we explained) on this platform, the average numerical error in $\cos^2 x + \sin^2 x - 1$ would be positive, possibly causing the secular error in total angular momentum. The asymmetric error on this platform is also seen in Fig. 5 (e), but the asymmetry is less conspicuous than in Fig. 5 (f) for IRIX 6.5 running MIPSpro C. This is because the histogram Fig. 5 (e) is calculated over the entire argument range of $x$, $[-\pi/128,\pi/128]$, which includes the range of $|x| \lesssim 0.01$ where numerical errors are distributed in quite a symmetric way.

The result from RedHat 7.1 on Alpha running Compaq C is quite interesting (Fig. 4(d)). On average, the numerical error of $\cos^2 x + \sin^2 x - 1$ on this platform seems symmetric along the $y = 0$ axis; mainly along $y = +1,0,-1$ and several points are seen on the lines $y = \pm 2$. However if you closely look at the lines $y = \pm 2$ on this panel, you will see that the errors are condensed and constitute a bunch of truncated "lines" with a typical length of 0.002—0.003. If we choose many values over the entire range of $x$ in this panel, $[-\pi/128,\pi/128]$, the average numerical error of $\cos^2 x + \sin^2 x - 1$ could be close to zero. However we have to notice that on this panel, the short "lines" are distributed out of phase on $y = +2$ and $y = -2$: if we take a look at a region of $x$ where there is one of these truncated "lines" along $y = +2$ (or $y = -2$), there is no point along the other side, $y = -2$ (or $y = +2$). Particularly at around $x \sim 0.0145$ with which we are concerned now, the error points of $\cos^2 x + \sin^2 x - 1$ are distributed along $y = -1,0,+1$ continuously and $y = +2$ as a short "line", which is not seen along $y = -2$ in this $x$ range. This condensation of the points along $y = +2$ raises the average numerical error of $\cos^2 x + \sin^2 x - 1$ in this $x$ range above zero. This positive average error of $\cos^2 x + \sin^2 x - 1$ on this platform can lead to the accumulation of the secular error in total angular momentum as we saw in Fig. 1(a). The asymmetric error on this platform is hardly seen in the histogram Fig. 5 (d) which has summed up the numerical errors over the entire range of $x$, $[-\pi/128,\pi/128]$.

In the left three panels (a)(b) and (c) of Fig. 4, numerical deviation of $\cos^2 x + \sin^2 x - 1$ from zero is distributed almost symmetrically along the line $y = 0$. Especially in the panels of Fig. 4 (a) of FreeBSD + GNU C and (b) of Solaris 2.5.1 + Fujitsu C, errors are concentrated only on the $y = -1,0,+1$ lines. So the average errors of $\cos^2 x + \sin^2 x - 1$ are expected to be nearly zero. This probably leads to the non-existence of secular error in angular momentum on these platforms as in Fig. 1(c) or (b).

**Table 1.** Major computer platforms we used for the accuracy tests described in this manuscript. VPP5000 (Fujitsu) is a VLIW-type RISC processor with vector pipelines. VPP300 (Fujitsu) is an LIW-type RISC processor with vector pipelines. "5.3R" of FreeBSD denotes "5.3—RELEASE". Some platforms and compilers in this table are no longer available or supported as of May 2005.

| OS | Compiler | Processor | Notation in Fig. | | | | |
|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 4,5 | 6 | 8 |
| FreeBSD 5.3R | gcc 3.3.3 | Pentium 4 | (c) | (c) | (a) | (d)(j) | (d) |
| RedHat 9 | Intel C 7.1 | Xeon | (f ) | (f ) | | | (f ) |
| UXP/V V20L10 | C V20L20 | VPP5000 | (l) | (l) | (c) | (e)(k) | |
| UXP/V V10L10 | C/VP V10L10 | VPP300 | (j) | | | | |
| Solaris 2.8 | Fujitsu C 5.1.1 | hyperSPARC | (a) | | (e) | (a)(g) | (b)(h) |
| Solaris 2.5.1 | Fujitsu C 4.0.2 | hyperSPARC | (b) | (b) | (b) | (b)(h) | (a)(g) |
| IRIX 6.5 | MIPSpro C 7.3.1 | R12000 | (d) | | (f ) | (f)(l) | |
| IRIX 6.2 | MIPSpro C 7.3 | Reality Engine | (e) | (e) | | | |
| RedHat 7.1 | Compaq C V6.4.9 | Alpha 21264 | (g) | | (d) | (c)(i) | (e) |
| HP–UX 10.20B | HP C A.10.32.03 | PA–RISC 7300 | (h) | (h) | | | |
| Super–UX R9.1 Rev. 1 | C/SX V2 Rev. 4.01 | SX–5 | (k) | | | | |
| HI–UX/MPP (03–00) | Optimized C (01–00) | SR8000 | (i) | (i) | | | |

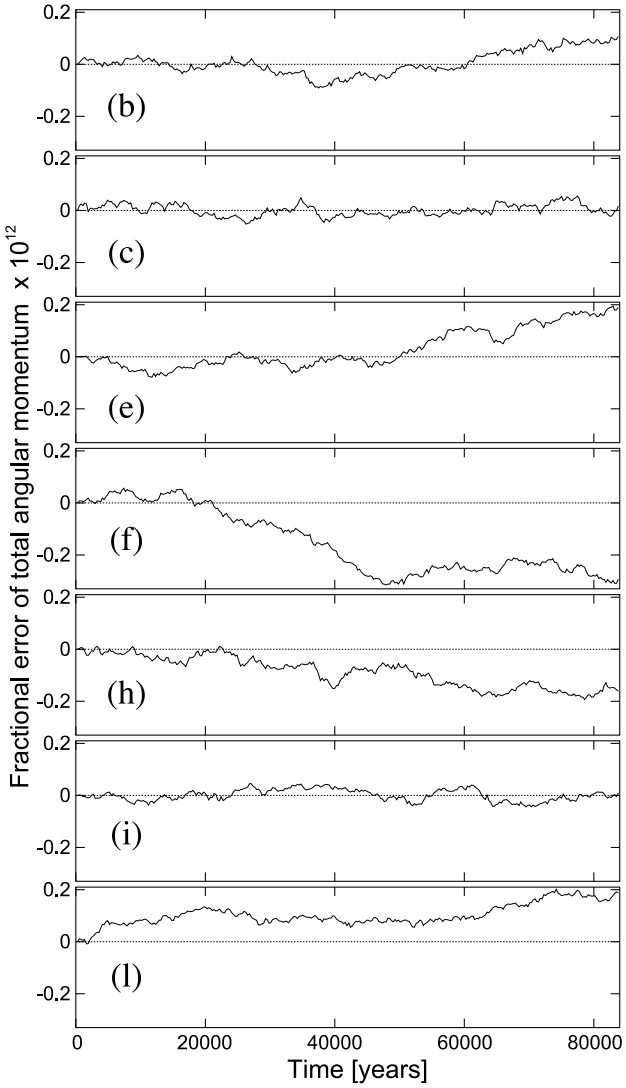



Fig. 2. Some of the panels in Fig. 1 whose numerical errors look much better than others are magnified: (b) Solaris 2.5.1 + Fujitsu C 4.0.2, (c) FreeBSD 5.3–RELEASE + GNU C 3.3.3, (e) IRIX 6.2 + MIPSpro C 7.3, (f) RedHat 9 + Intel C 7.1, (h) HP–UX 10.20B + HP C A.10.32.03, (i) HI–UX/MPP + Optimized C on Hitachi SR8000, and (l) UXP/V V20L10 + C V20L20 on Fujitsu VPP5000. Notation of the panels is the same as in Fig. 1. The vertical scale is magnified $10^{12}$ times. See Table 1 for more detail.

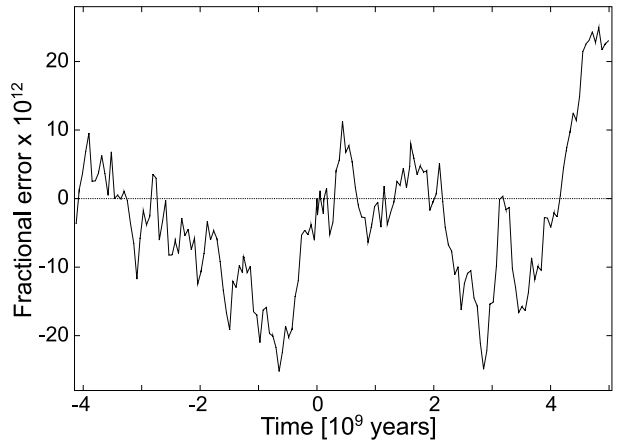



Fig. 3. Numerical error of total angular momentum in a nine-planet system from one of the very long-term integrations by Ito & Tanikawa (2002) performed mainly on a Solaris 2.5.1 platform using Fujitsu C. The vertical scale is magnified $10^{12}$ times.

In Fig. 4(c) that is for the new UXP/V on VPP5000, we see another interesting phenomenon, though it is not directly related to the numerical inaccuracy discussed in this manuscript. The result from this platform seems symmetric around $y = 0$, but includes many $\cos^2 x + \sin^2 x - 1$ values that are obviously smaller than the machine epsilon of IEEE 754 double precision arithmetic. This is due to the so-called "multiply add/subtract instruction" particular to the VPP5000 CPU. In this CPU, an operation containing multiplication as well as addition/subtraction can be optimized, being performed by a single hardware instruction. During the optimized operation, results of multiplications (such as $\cos^2 x$ or $\sin^2 x$) are added/subtracted before being rounded. This is why the numerical errors shown in Fig. 4(c) are sometimes smaller than the machine epsilon. In addition to the VPP5000 CPU, Intel Itanium-2 has a similar "multiply add/subtract instruction" architecture, where numerical errors of arithmetic operations can sometimes be smaller than those from conventional processors.

Now we think the accuracy test exhibited in Fig. 4 has revealed many of the causes of the numerical errors shown in Fig. 1. A remaining problem might be that the range of argument $x$ is quite small in Fig. 4, only from $-\pi/128$ to $\pi/128$. This is because amount of output data this test creates is so large ($\sim$ 2 Gbyte per test) due to the small stepsize we used ($2^{-27}$) that we cannot go through a very large argument range, such as $[0,2\pi]$. To inspect a wide argument range, we performed another kind of accuracy test for `sin()` and `cos()`. We calculated `sin()` and `cos()` using both double precision arithmetic (hereafter denoted by the subscript "d") and extended-double (quadrupole) precision arithmetic (hereafter denoted by the subscript "q", computed in a Fortran (`real*16`) environment) over a wider range of $x$ argument, $[0,4\pi]$, with a larger stepsize of $2^{-16}$. Then we measured their difference, assuming the result from the quadrupole precision arithmetic is closer to the true values of $\sin x$ or $\cos x$. To further reduce the amount of data, we sum up every 1024 results in our calculation. In summary, what we have calculated is a bunch of pair values

$$\delta \sin x \equiv \sum_{i=1}^{N} (\sin_{\mathrm{d}} x_i - \sin_{\mathrm{q}} x_i),$$
$$\delta \cos x \equiv \sum_{i=1}^{N} (\cos_{\mathrm{d}} x_i - \cos_{\mathrm{q}} x_i) \qquad (7)$$

for a series of $x_1$ as $x_1 = 0$, $N\Delta x$, $2N\Delta x$, ..., where $N = 1024$ and $\Delta x \equiv x_{i+1} - x_i = 2^{-16}$, until $x_1$ reaches $4\pi$. The result of this test is summarized in Fig. 6. We chose six platforms that are listed in one of the right-hand side columns of Table 1. As expected, Solaris 2.5.1 running Fujitsu C (Fig. 6(b) and (h)) and FreeBSD 5.3-RELEASE running GNU C (Fig. 6(d) and (j)), neither of which indicated a secular error in the conservation of total angular momentum in Fig. 1, yield smaller errors than others do, $\sim 10^{-15}$. Results from the Solaris 2.8 platform running Fujitsu C (Fig. 6(a) and (g)) shows larger errors compared with those of Solaris 2.5.1 and FreeBSD, which possibly are related to the larger numerical errors that this platform has shown in Fig. 1(a) and Fig. 4(e).

Interestingly, RedHat Linux on Alpha running Compaq C (Fig. 6 (c) and (i)), which turned out to be equipped with

inaccurate numerical trigonometric subroutines through our previous tests, now shows better behavior in terms of $\delta \sin x$ and $\delta \cos x$. The error is around $\sim 10^{-15}$ , smaller than that from the old Solaris platform (Fig. 6 (b) and (h)), and even as small as that from the FreeBSD platform (Fig. 6 (d) and (j)). Nevertheless, as we have seen, this platform yields a substantial error in the total angular momentum conservation as in Fig. 1(a), probably through the fine asymmetric structure of numerical errors as shown in Fig. 4(d). This apparently good behavior appeared because the fine structure of the numerical errors of `sin()` and `cos()` that we saw in Fig. 4(d) are blurred by the averaging procedure over every 1024 data points that we have done to draw Fig. 6 (c) and (i). Hence the apparently good behavior of this platform indicates in Fig. 6(c)(i) does not necessarily prove that this platform is equipped with a very good numerical accuracy.

Note that in the panels for Fujitsu UXP/V on VPP5000 (Fig. 6(e) and (k)) and SGI IRIX 6.5 (Fig. 6(f) and (l)), vertical scale is about an order larger than in the panels for other platforms. This fact clearly demonstrates that these two platforms have significantly larger numerical inaccuracies (i.e. $\delta \sin x$ and $\delta \cos x$) compared with other platforms. Especially IRIX 6.5, which has shown an obvious secular error in total angular momentum conservation in Fig. 1(d) and an asymmetric error distribution in Fig. 4(f), exhibits huge $\delta \sin x$ and $\delta \cos x$ in panels (f) and (l) of Fig. 6. Fujitsu UXP/V did not show significantly large errors in previous accuracy tests (Fig. 1(l) and Fig. 4(c)) in the limited range of trigonometric argument. But looking at the large $\delta \sin x$ and $\delta \cos x$ in Fig. 6(e) and (k), we cannot be completely sure whether this platform will maintain numerical results as accurate as it showed in Fig. 1(l) and in Fig. 4(c) when we choose a wider range of trigonometric arguments, for example, ones that give a considerable number of errors in Fig. 6(e) and (k), such as $x \sim 0.7\pi$ when $\delta \sin x > 1.5 \times 10^{-14}$ (Fig. 6(e)).

Remember that among the secular numerical errors in total angular momentum shown in Fig. 1, that from the old UXP/V (V10L10) on Fujitsu VPP300 running C/VP exhibited far bigger, an error $10^3$ to $10^4$ times larger than those produced on other platforms (Fig. 1(j)). Since the old Fujitsu VPP300 platform had been replaced to a new one (Fujitsu VPP5000 with UXP/V V20L10) and gone before we performed the series of accuracy tests as in Fig. 4, we cannot know what the numerical errors on VPP300 would have been if we performed the same accuracy test on this platform. But fortunately, we had calculated $\delta \sin x$ on the VPP300 platform before the replacement, whose result is shown as Fig. 7. As we expected, $\delta \sin x$ on this platform is about two orders of magnitude larger than other "well

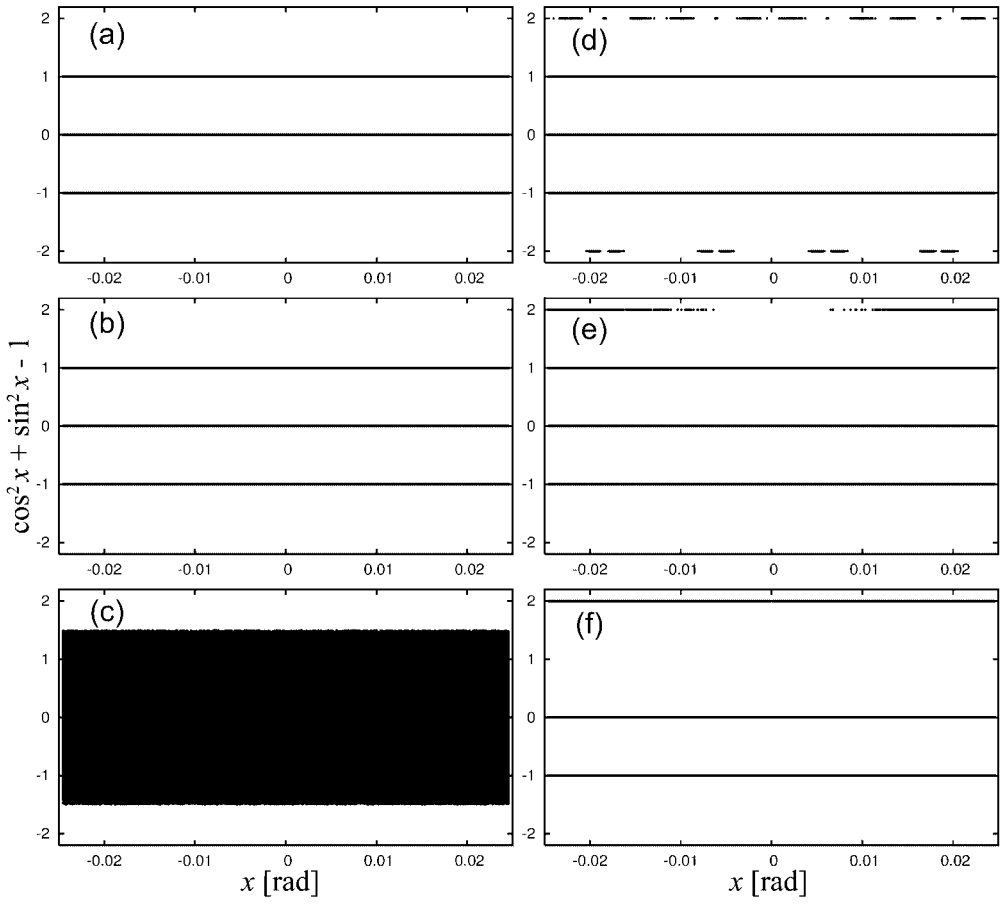



Fig. 4. Residual of $\cos^2 x + \sin^2 x - 1$ normalized by the double-precision arithmetic machine epsilon, $\varepsilon \sim 1.1102230246251565 \times 10^{-16}$ . (a) FreeBSD 5.3–RELEASE + GNU C 3.3.3, (b) Solaris 2.5.1 + Fujitsu C 4.0.2, (c) UXP/V V20L10 + C V20L20 on Fujitsu VPP5000, (d) RedHat 7.1 + Compaq C V6.4.9 on Alpha, (e) Solaris 2.8 + Fujitsu C 5.1.1, (f) IRIX 6.5 + MIPSpro C 7.3.1. The unit of $x$ is radian, and the $x$ range is $[-\pi/128, \pi/128]$ which is close to [−0.02454, +0.02454].

behaved" platforms like Solaris 2.5.1 or FreeBSD. As we mention again later, the large error of this platform was caused by its rounding mode which was not toward the nearest as on most computer platforms, but toward 0 for reasons of hardware. Though now we cannot reproduce the data shown in Fig. 7 because the VPP300 platform is no longer available, the result of $\cos^2 x + \sin^2 x - 1$ on this platform would have been quite asymmetric along $y = 0$ axis, if plotted on a figure like Fig. 4, perhaps much stranger than what are shown in Fig. 4(d), (e), or (f).

## 4. Measures to prevent numerical errors

Ever since we found the numerical inaccuracies and understood their probable cause, we have been trying to solve the problem, taking measures to protect clients of our computer systems against encountering this kind of problem in their numerical calculations.

What we did first was to look for a description of the nominal accuracy of mathematical libraries provided by computer vendors. Some documents turned out to have descriptions of this issue (see Appendix A.3 for examples). But since the actual implementation of mathematical functions such as `sin()` and `cos()` are not specified in IEEE 754 floating point formats, we thought there was no guarantee that the descriptions of the function accuracy (as in Appendix A.3) were absolutely true; we just had to believe what computer vendors told us. Next, we reported the existence of the numerical inaccuracies to the vendors of each computer platform to ask their opinions. Responses from computer vendors differed vendor by vendor.

### *4.1 Responses from computer vendors*

#### *4.1.1 Sun Microsystems (Solaris)*

This vendor admitted that some trigonometric functions such as `sin()` and `cos()` in their mathematical library bundled with the newer Solaris (2.6 and later) use different algorithms from the previous version: The previous version of `sin()` and `cos()` on Solaris 2.5.1 used a 13th-order polynomial approximation to obtain the values of these functions, but the newer version employs a table-lookup method. They said that the difference between old and new `sin()` in double precision arithmetic was only the last bit in fraction, and that it was actually out of the double precision digits and did not have an accuracy problem. However, this vendor guaranteed that we could use the mathematical library of Solaris 2.5.1 (hereafter called `libm251.a`) which had greater accuracy, at least with respect to `sin()` and `cos()`. The vendor promised that they would officially support the old math library, `libm251.a`, if we encountered any inconvenience or troubles when using it. So we copied `libm251.a` on all the new Solaris nodes and announced to our clients that they could use this library by using the link option `-lm251` when they link their object

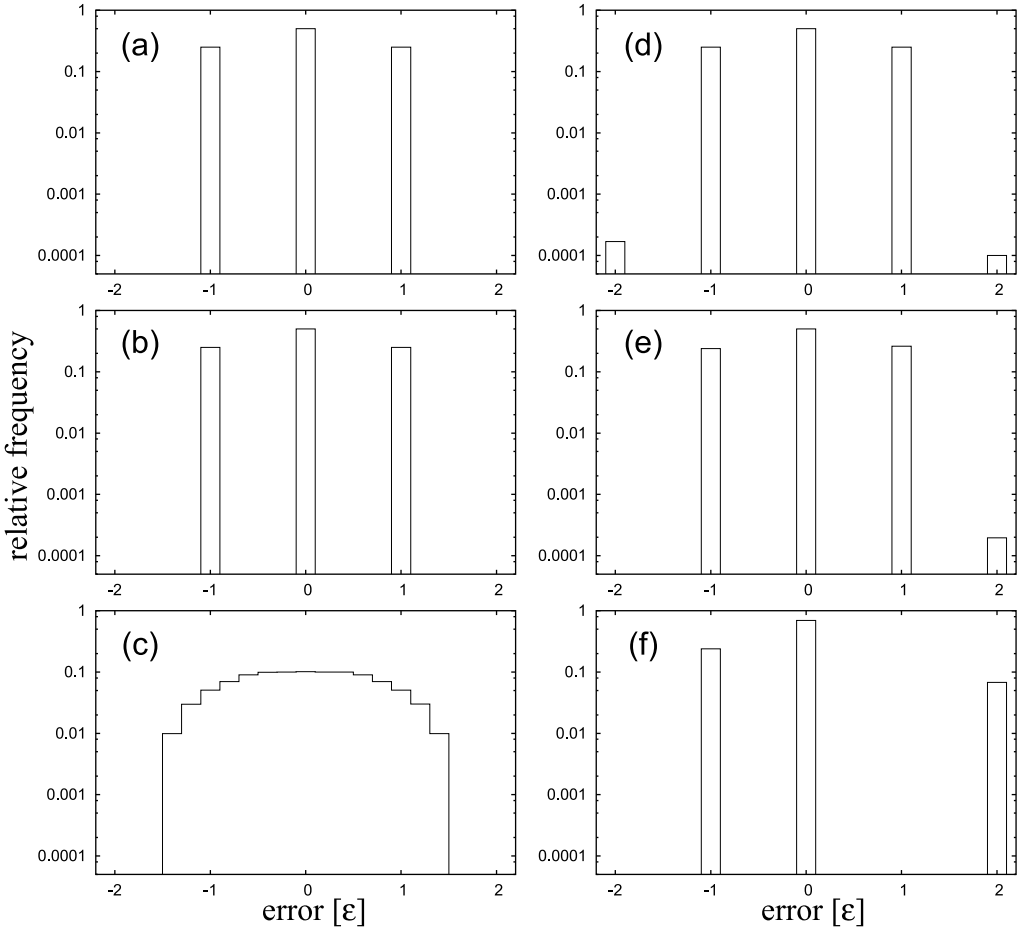



Fig. 5. Histograms for the relative frequency of each error value described in Fig. 4. Notation of the panels is the same as in Fig. 4. The horizontal unit is the double-precision arithmetic machine epsilon, $\varepsilon \sim 1.1102230246251565 \times 10^{-16}$ .

modules including `sin()`, `cos()`, and other mathematical functions whose newer Solaris version might contain numerical inaccuracies.

*4.1.2 Fujitsu (VPP)*

The numerical error that we saw on the old UXP/V (V10L10) platform (Fig. 1(j) and Fig. 7) is far larger than on any other platforms. When we asked the vendor about this problem in 1999, the vendor explained that this was caused by the fact that the rounding mode on this platform was not toward the nearest but toward 0 for hardware reasons, largely for computation speed. At that time we did not further investigate the error source on this platform because we had been planning to replace this old platform (VPP300) with a new one, hopefully with new mathematical libraries with a better accuracy. As we had hoped, the new system (UXP/V V20L10 on VPP5000) exhibits better accuracy (Fig. 1(l)) than the old one, together with having a rounding mode toward the nearest. But as we have checked the accuracy of mathematical functions on the new platform in detail, it has turned out that this platform might still contain some non-negligible numerical error in some part of its mathematical functions as we discussed around Fig. 6(e) and (k).

*4.1.3 HIT (Alpha), Visual Technology (Alpha), and SGI Japan (IRIX)*

For Alpha (Tru64 or RedHat Linux) and IRIX platforms that exhibited substantial numerical inaccuracy shown

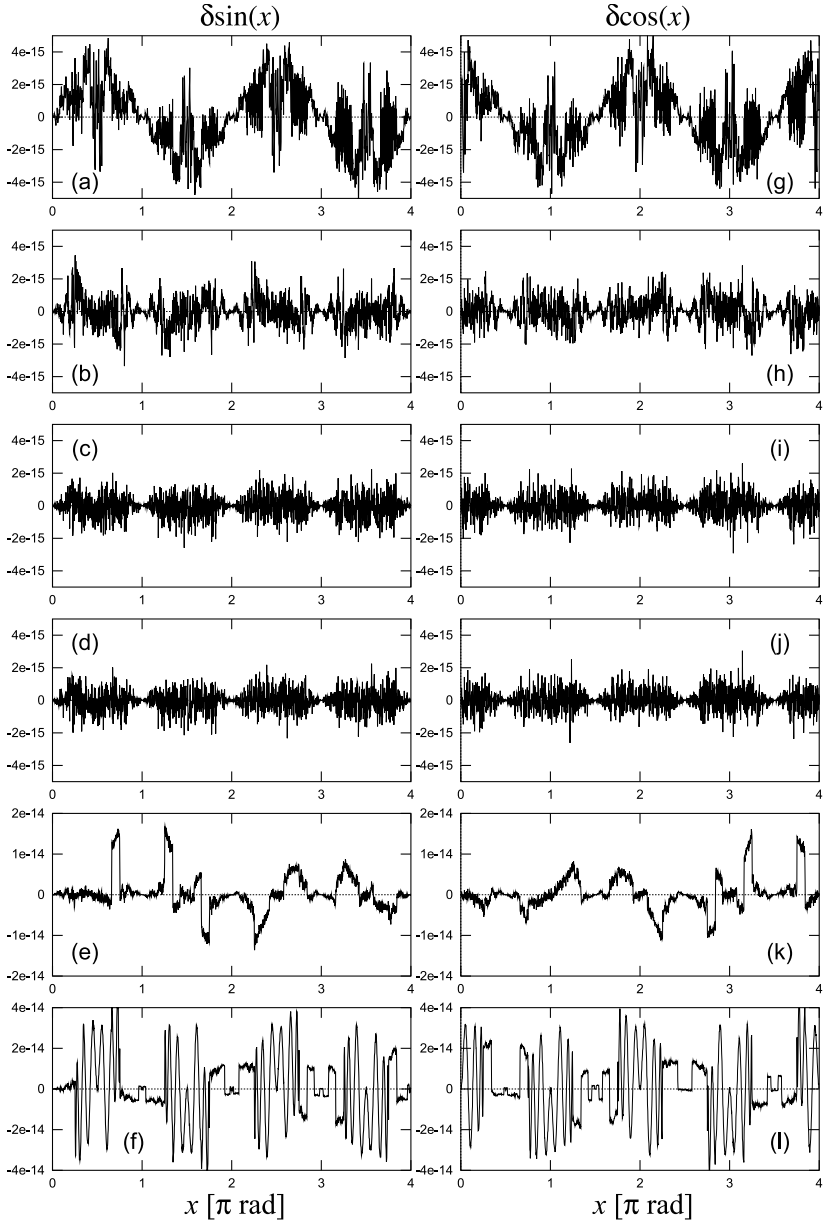



Fig. 6. $\delta \sin x$ (left panels) and $\delta \cos x$ (right panels) defined by equation (7). (a)(g) Solaris 2.8 + Fujitsu C 5.1.1, (b)(h) Solaris 2.5.1 + Fujitsu C 4.0.2, (c)(i) RedHat 7.1 + Compaq C V6.4.9 on Alpha, (d)(j) FreeBSD 5.3–RELEASE + GNU C 3.3.3, (e)(k) UXP/V V20L10 + C V20L20 on Fujitsu VPP5000, (f)(l) IRIX 6.5 + MIPSpro C 7.3.1. The unit of $x$ is $\pi$ radian. Note that the vertical range of panels (e) and (k) is about five times larger than those of (a)–(d) and (g)–(j), and that of panels (f) and (l) is about two times larger than that of (e) and (k).

in Fig. 1(a) and (d), we first asked their vendors (HIT for Tru64 on Alpha, Visual Technology for Alpha RedHat Linux, and SGI for IRIX) if it was possible to modify the default mathematical libraries. But it turned out that modifying the existing mathematical library would be a formidable task in terms of both time and cost. In the end, these vendors decided to port a free mathematical library, libm in GNU libc (hereafter called GNU libm and GNU libc, respectively). We modified the source code of part of the GNU libm, compiled the source code with optimization, made a library, and let our customers use it. The resulting library exhibits satisfactory accuracy for our purpose. In the next subsection we describe the process that we took to port and use the GNU libm on our computer platforms.

*4.2 Porting and using GNU libm*

We had been aware that the GNU libm in general yielded good accuracy when it was used with GNU C (`gcc`). However, for example on our Alpha platforms, the default (= vender-recommended) C compiler was Compaq C, which uses a dedicated math library called CPML (Compaq Portable Math Library). According to its webpage (`http://h18000.www1.hp.com/math/`), CPML is "a collection of fast, reliable, highly accurate routines that support a wide variety of mathematical functions across popular operating systems, hardware architectures, and languages." But as we have seen in previous sections, the accuracy of CPML with Compaq C on Alpha seems to have a non-negligible problem, as does the MIPSpro C compiler on our SGI IRIX platform (IRIX 6.4 or later).

Our first solution to the accuracy problem on Alpha as well as on the IRIX platform was to port, optimize, and install the GNU libm so that we can use it with `gcc`, which is supposed to match the GNU libm quite well. Our procedure to make an optimized GNU libm was roughly as follows: There might be a better way to achieve our purpose, but what we did and how we did it is, we think, quite straightforward and easy to implement.

(1) Select the part of the source code of the GNU libc (`glibc`; we used version 2.1.2 at that time) that is responsible for mathematical functions including `sin()` and `cos()`, which we have focused on so far.

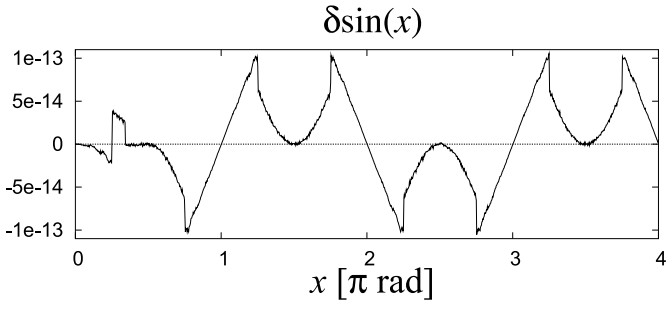



Fig. 7. $\delta \sin x$ defined by equation (7) which was calculated on UXP/V V10L10 with C/VP V10L10 on Fujitsu VPP300. The unit of $x$ is $\pi$ radian. Note that the vertical range is about 25 times larger than those of panels (a)–(d) and (g)–(j) in Fig. 6.

(2) Modify the source code so that we can recompile it on the target platform. Modifications are often necessary around specific macros called `weak_alias` or `strong_alias` in `glibc` which have the role of absorbing the difference in interpreting names of variables on different platforms.

(3) Recompile the modified source code by `gcc` with appropriate optimized options such as `-O3`, and make a bunch of relevant object modules `*.o`. The default optimization option of `glibc` (2.1.2) is just `-O`, not `-O3` or `-O4`, for some reason that we do not know.

(4) Archive all the object modules by `ar`, bundling them into a single static library such as `libm_opt.a`. Then we can use the optimized GNU libm as usual with `gcc` such as

```
gcc source.c -lmgcc_opt
```

As we tested `libm_opt.a` in our Alpha environment, it worked quite well as to accuracy, yielding a numerical error as small as those on FreeBSD platforms. However, the performance (= computation speed) of the GNU math library with `gcc` is not as good as that of Compaq C with CPML: The performance rate between Compaq C with CPML and `gcc` with our GNU libm was about 1.2—1.8, depending on the program we ran. We installed the GNU libm not only on our Alpha platforms but on our SGI IRIX platform where there were also no accurate trigonometric function provided by the operating system. Though the optimized GNU libm again worked quite well as to accuracy (as good as that of FreeBSD platforms), we were not satisfied with the computation speed: The performance rate between MIPSpro C and `gcc` with the optimized GNU libm is around 1.5. From the standpoint of compiler optimization on local computer platforms, this performance difference might be reasonable and rather obvious because commercial compilers and mathematical libraries are usually much better optimized and localized for a particular hardware platform (such as Compaq C with CMPL for Alpha processors, MIPSpro Compiler for MIPS processors, and Intel Compilers for Intel processors) than is generic freeware such as GNU products. Aside from potential accuracy problems in these commercial compiling environments, we should make use of their high performance in our numerical calculations.

What we gave some thoughts to then was: the mathematical functions that we have so far found to have obvious accuracy problems are `sin()` and `cos()`. Though there may be other mathematical subroutines that contain significant inaccuracy, for now it might be okay for us to use only `sin()` and `cos()` among the ported GNU libm functions, calling all the other mathematical subroutines from the default library that is bundled with the operating system or provided with commercial compilers. If in future we find a problem in any mathematical functions other than `sin()` and `cos()` in the default library, we would just have to replace the troubled function in the default library for those in the GNU libm.

In summary, here is what we have done for this purpose:

(1) Rebuild specific mathematical functions in the GNU libm such as `sin()` or `cos()` with appropriate optimization as we illustrated.

(2) Note that we have to rename the troubled functions in the GNU library source code such as `sin()` to `glib_sin()`, and `cos()` to `glib_cos()`. These will be the aliases for these functions in the source code that we use. If you do not rename these functions, they will be in conflict with the functions with the same name in the os-default (or compiler-bundled) mathematical library that we might link later.

(3) According to the procedure that we described before, make a library archive such as `libm_opt.a`.

(4) Make a C header file in order to make the above aliases available in our source programs. For example,

```
extern double __glib_sin(double);
#define sin __glib_sin
extern double __glib_cos(double);
#define cos __glib_cos
```

(5) Suppose the C header is named *glib_math.h*. Include this file wherever necessary in source code such as

```
#include <math.h>
#include <glib_math.h>
```

This method principally enables us to deploy particular mathematical functions (such as `sin()` or `cos()`) from particular libraries (such as `libm_opt.a`) that we specify when linking. As a compiler, we can use whatever we like, such as Compaq C (`ccc`) on Alpha or MIPSpro C on SGI IRIX. For example, a link option using Compaq C with `libm_opt.a` will be like

```
ccc source.c -lm_opt -lm
```

In this example, only `sin()` and `cos()` are linked to the source code from `libm_opt.a` through the `-lm_opt` option, while other mathematical functions will all be consulted by the default library through the `-lm` option. The result of these measures is excellent as expected, achieving satisfactory accuracy of the GNU libm, as well as high performance of commercial compilers and libraries.

Ever since we discovered the existence of the numerical inaccuracies in `sin()` and `cos()` on certain computer platforms, we have cautioned our clients and alerted them to the potential inaccuracy of those functions. Our announcement about this problem is available through e-mails and our webpage, mainly in the form of FAQ (frequently asked/answered questions). Also, we have summarized how we optimized and installed the GNU libm in a brief document, which we can share on request with any readers that are interested in and are trying to solve this kind of problem.

## 5. Discussion

### *5.1 Further improvement of mathematical subroutines*

As we mentioned first, the investigation of numerical inaccuracies of mathematical functions that we have described in this manuscript has just begun, and is still far from completed. Sooner or later we will have to inspect potential numerical inaccuracy that could be hidden in many other math functions; for instance, other trigonometric functions such as `tan()`, exponential and logarithmic functions such as `exp()` or `log()`, or inverse trigonometric functions such as `asin()`. Then we will need to check out, not only the C language environment, but the Fortran environment that has been extensively used by numerical computing scientists for many decades. We should also check the accuracy of mathematical functions not only of double-precision but of higher precision, such as quadrupole precision. We have to continue performing accuracy tests on as many kinds of platform as possible whenever we introduce a new computer system with a new hardware/os/compiler environment, such as the 64-bit processor like IA–64 or Opteron.

As for the secular numerical error in angular momentum described in Section 2, the most fundamental solution would be to avoid using trigonometric functions when we solve Kepler's equation, and to use more sophisticated algorithms such as what T. Fukushima has proposed (Fukushima, 1997). But even if we know there are new methods that are better and more desirable for this purpose, we cannot prevent our clients from using their own favorite method (even if that is really obsolete!). For example, in solving Kepler's equation, using a generic Newton's method with trigonometric functions is quite popular due to its easy implementation. Probably a certain number of researchers still use it. Also, there are many other numerical problems that inevitably involve transcendental functions. Therefore detailed investigation of computer math libraries in this manuscript is justified, and we believe it will be more important and necessary in the near future.

Talking about future work, there is another direction to go along this line: improving the algorithms themselves of mathematical subroutines that possibly yield inaccurate values. In general, computer vendors keep the algorithms of their mathematical subroutines classified. But it is not difficult to imagine that `sin()` and `cos()` functions in most computer math libraries are based on Taylor expansion series such as

$$\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \cdots, \quad (8)$$

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} = 1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720} - \cdots, \quad (9)$$

Using this method, we might need to include a large number of higher-order terms to maintain accuracy of approximation, especially when the absolute value of $x$ is large. However, incorporation of higher-order terms could significantly increase the computation time. This is the point at which computer vendors should devise some trick to achieve high eciency when calculating transcendental functions, keeping accuracy intact. For example, an algorithm to

calculate `sin()` and `cos()` is open to the public in the GNU libc, `glibc`. `glibc` uses a polynomial of degree 14 for `cos()` and that of degree 13 for `sin()` on $[0,\pi/4]$. Based on this polynomial expansion, *glibc* adds a small but smart modification for better accuracy, exploiting the addition theorem of trigonometric functions (see `README.libm` in the `glibc` source package for technical details). Judging from the fact that the GNU libm has given us very good accuracy in tests, the algorithms implemented in the GNU libc seem to score a greater deal, at least in normal use by numerical astrophysists, than other algorithms provided by many computer vendors. Implementing this kind of new technique as a local tool and providing it to our clients might be a direction our computer center takes in the near future, if the new routines are faster and more accurate than the existing math libraries, and if they are easily implemented.

There are already several studies along this line. For example, if we are using the seventh-order Taylor expansion polynomial for approximating sin $x$ within the accuracy of single precision arithmetic as

$$\begin{aligned}\sin x &= x-\frac{x^3}{6}+\frac{x^5}{120}-\frac{x^7}{5040}+\cdots,\\ &\simeq x-0.166666667x^3+0.0083333333x^5\\ &\quad -0.000198412698x^7 \end{aligned} \tag{10}$$

Green (2003) found that we could obtain a much better solution just by adding a slight modification to the coeffcients in polynomial (10) through the concept of minimax polynomials as

$$\begin{aligned}\sin x &\simeq x-0.166666567x^3+0.0083322080x^5\\ &\quad -0.000195168955x^7 \end{aligned} \tag{11}$$

In principle, we can apply this kind of method to many

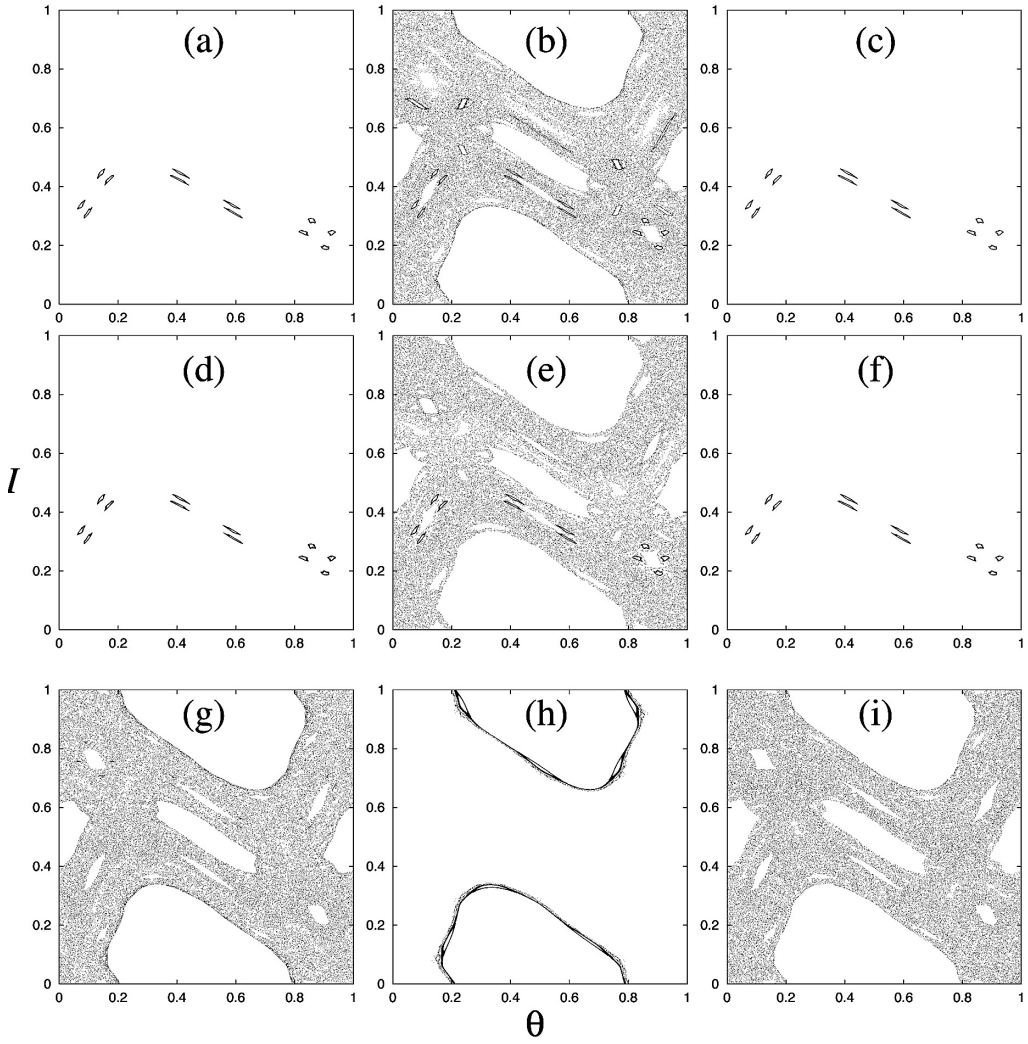



Fig. 8. Two sets of surfaces of sections by the standard map defined by equations (12). 50,000 points are plotted on each panel. The first set consists of six panels, starting from ($\theta_0$, $I_0$) = (0.90002, 0.19998): (a) Solaris 2.5.1 + Fujitsu C 4.0.2, (b) Solaris 2.8 + Fujitsu C 5.1.1, (c) Solaris 2.8 + Sun WorkShop 6 Fortran 77 `(real*16)`, (d) FreeBSD 5.3–RELEASE + GNU C 3.3.3, (e) RedHat 7.1 + Compaq C V6.4.9 on Alpha, and (f) RedHat 9 + Intel C 7.1. The second set consists of three panels, starting from ($\theta_0$, $I_0$) = (0.56261, 0.68744): (g) Solaris 2.5.1 + Fujitsu C 4.0.2, (h) Solaris 2.8 + Fujitsu C 5.1.1, and (i) Solaris 2.8 + Sun WorkShop 6 Fortran 77 `(real*16)`. Note that the results in panels (c) and (i) are calculated by quadruple-precision arithmetic implemented in Fortran language `(real*16)` in order to get a solution with higher accuracy.

other numerical transcendental subroutines and in other arithmetic precisions as well.

*5.2 Numerical error in chaos systems*

At the end of this paper, let us show an example where a very slight numerical error in a mathematical function can be substantially magnified, ending up with a completely different solution.

A certain kind of numerical computation inevitably and frequently deploys transcendental functions. A typical example is in the field of dynamical systems research. A convenient way to study phase space trajectories, particularly in problems with two degrees of freedom, is to look at a surface of section of phase space using a two-dimensional map. Among numerous sorts of map, the standard map has been studied for many decades for its simple form as well as its deep significance in the field of nonlinear dynamics, especially in that of chaos. The standard map, also known as the Chirikov–Taylor map (Lichtenberg and Lieberman, 1992), is expressed by the $n$-th action $I_n$ and angle $\theta_n$ as

$$\begin{cases} I_{n+1} = I_n + K\sin\theta_n, \\ \theta_{n+1} = \theta_n + I_{n+1} \end{cases} \qquad (12)$$

where $K$ is the stochasticity parameter which controls the degree of chaos of the system. Since the map (12) is constructed on the calculation of sin $\theta_n$, we cannot avoid very frequent use of `sin()`.

Depending on $K$ and initial starting point ($\theta_0$, $I_0$), the points that this map produces on the surface of section of phase space can be distributed in a totally different way, even when we start from a similar initial point. This is a typical behavior of chaos: high sensitivity to initial conditions. In this kind of map, a very small error in numerical mathematical functions such as `sin()` can be significantly magnified, totally changing the numerical solution.

Fig. 8 shows examples of surface of section produced by map (12), starting from a pair of initial points. First, each of the six panels (a)—(f) in Fig. 8 shows 50,000 points starting at an initial point ($\theta_0$, $I_0$) = (0.90002, 0.19998) on several different computer platforms. Among the six panels, what we believe to be the closest to the true solution is (c), which was calculated with quadruple precision (`real*16`) in Fortran. The results in others (the panels (a)(b)(d)(e) and (f)) were calculated with double precision in C. Platforms that yield solutions closer to that of the quadruple precision calculation (c) are: (a) Solaris 2.5.1 with Fujitsu C, (d) GNU C on FreeBSD, and (f) Intel C on RedHat Linux on Xeon. The points on these three panels are distributed rather regularly, being confined to the edges of stable islands. These three platforms have indicated better accuracy in our previous numerical tests. On the other hand, the results from Solaris 2.8 running Fujitsu C (b) and Alpha Linux running Compaq C (e) are very different from others: the points are all scattered on the vast sea of chaos. These two platforms have turned out to have accuracy problems, as described in the previous sections.

This kind of clear difference is also seen at some other initial positions. Each of the three panels (g)—(i) in Fig. 8 shows 50,000 points starting from ($\theta_0$, $I_0$) = (0.56261, 0.68744). What we believe to be the closest to the true solution in this case is (i), which was calculated with quadruple precision in Fortran, whose points are widely scattered over the chaos sea. The points generated by the Solaris 2.5.1 platform (g) again show good agreement with (i), while the points produced by the new Solaris 2.8 platform (h) behave differently, sticking to the border between regular solutions and the sea of chaos.

The examples in Fig. 8 are some of the most typical of those that exhibit acute differences in numerical results. However, since the number of initial points in this phase space that we have inspected for this test is limited, there must be many more initial points that would produce groups of points whose behavior differ depending on the computer platform we use. These kinds of chaos systems exquisitely illustrate the necessity to check and improve the accuracy of numerical mathematical functions, a task that very few computer vendors are willing to undertake.

**Acknowledgments**

The authors have benefited from useful information from many people, especially from Norimichi Suzuki (HIT), Atsushi Kawamata (Visual Technology), and some anonymous engineers at Fujitsu, SGI Japan, and Sun Microsystems. Kouji Hamakawa and Chie Naitou (EXEC) have provided and maintained excellent proving grounds for the accuracy tests described in this manuscript. Detailed and constructive review by Yolande McLean has, as usual, considerably improved the English presentation of this paper. The referee also suggested directions which bettered the quality of this paper.

**Appendix**

Here we briefly summarize the concept of floating-point formats (Section A.1) and its current standard, IEEE 754 (Section A.2), according to Sun Microsystems' *Numerical Computational Guide* (see Section A.3 for the relevant URL). We also mention the nominal accuracy of numerical transcendental functions on several computer platforms (Section A.3).

*A.1 Floating-point formats*

Floating-point representations have a base $\beta$ (which is always assumed to be even) and a precision $p$. In general, a floating-point number will be represented as $\pm d.dd\ldots d\times\beta^e$, where $d.dd\ldots d$ is called the "significand" (or "mantissa") and has $p$ digits. More precisely, $\pm d_0.d_1d_2\ldots d_{p-1}\times\beta^e$ represents the number

$$\pm\left(d_0 + d_1\beta^{-1} + \ldots + d_{p-1}\beta^{-(p-1)}\right)\beta^e,\ 0 \le d_i < \beta. \qquad (A.1)$$

The term "floating-point number" is used to mean a real number that can be exactly represented in the format under discussion. Two other parameters associated with floating-point representations are the largest and smallest allowable exponents, $e_{\max}$ and $e_{\min}$. Since there are $\beta^p$ possible significands, and $e_{\max}-e_{\max}+1$ possible exponents, a floating-point

number can be encoded in $\log_2 (e_{max} - e_{max} + 1) + \log_2 \beta^p + 1$ bits, where the final +1 is for the sign bit.

There are a few reasons why a real number might not be exactly representable as a floating-point number. The most common situation is illustrated by the decimal number 0.1. Although it has a finite decimal representation, in binary, it has an infinite repeating representation. Thus, when $\beta = 2$, the number 0.1 lies strictly between two floating-point numbers and is not exactly representable by either of them.

Since rounding errors are inherent in floating-point computation, it is important to have a way to measure these errors. As an example, consider the floating-point format with $\beta = 10$ and $p = 3$. If the result of a floating-point computation is $3.12 \times 10^{-2}$, and the answer when computed to infinite precision is 0.0314, it is clear that this is in error by 2 units in the last place. Similarly, if the real number 0.0314159 is represented as $3.14 \times 10^{-2}$, then it is in error by 0.159 units in the last place. In general, if the floating-point number $d.d \ldots d\beta^e$ is used to represent $z$, then it is in error by $|d.d \ldots d \quad (z/\beta^e)|\beta^{p-1}$ units in the last place. The term "ulps" is used as shorthand for "units in the last place." If the result of a calculation is the floating-point number nearest to the correct result, it still might be in error by as much as 0.5 ulp.

Another way to measure the difference between a floating-point number and the real number it is approximating is relative error, which is simply the difference between the two numbers divided by the real number. To compute the relative error that corresponds to 0.5 ulp, observe that when a real number is approximated by the closest possible floating-point number $d.dd \ldots dd \times \beta^e$, the error can be as large as $0.00...00\beta' \times \beta^e$, where $\beta'$ is the digit $\beta/2$, there are $p$ units in the significand of the floating-point number, and $p$ units of 0 in the significand of the error. This error is $((\beta/2)\beta^{-p}) \times \beta^e$. Since numbers of the form $d.dd \ldots dd \times \beta^e$ all have the same absolute error, but have values that range between $\beta^e$ and $\beta \times \beta^e$, the relative error ranges between $((\beta/2)\beta^{-p}) \times \beta^e/\beta^e$ and $((\beta/2)\beta^{-p}) \times \beta^e/\beta^{e+1}$. That is,

$$\frac{1}{2}\beta^{-p} \leq \frac{1}{2}\text{ulp} \leq \frac{\beta}{2}\beta^{-p}. \tag{A.2}$$

In particular, the relative error corresponding to 0.5 ulp can vary by a factor of $\beta$. Setting $\varepsilon = (\beta/2)\beta^{-p}$ to the largest of the bounds in equation (A.2), we can say that when a real number is rounded to the closest floating-point number, the relative error is always bounded by $\varepsilon$, which is referred to as "machine epsilon". Machine epsilon of the IEEE 754 single format ($\beta = 2$, $p = 24$) is about $5.9604645 \times 10^{-8}$, and that of double format ($\beta = 2$, $p = 53$) is about $1.1102230246251565 \times 10^{-16}$.

**Table A.1.** IEEE 754 format parameters and language types for single- and double-precision arithmetic. Base $\beta = 2$.

| Parameter | single | double |
|---|---|---|
| $p$ | 24 | 53 |
| $e_{min}$ | –126 | –1022 |
| $e_{max}$ | +127 | +1022 |
| Exponent width in bits | 8 | 11 |
| Format width in bits | 32 | 64 |
| **Language** | **Implementation** | |
| C, C++ | `float` | `double` |
| Fortran | `read*4` | `real*8` |

*A.2 IEEE 754*

IEEE 754 is the arithmetic model specified by the ANSI/IEEE Standard 754–1985 for Binary Floating-Point Arithmetic. IEEE 754 requires $\beta = 2$, and specifies two basic floating-point formats: single and double (Table A.1). The IEEE single format has a significand precision of 24 bits ($p = 24$) and occupies 32 bits overall. The IEEE double format has a significand precision of 53 bits ($p = 53$) and occupies 64 bits overall. IEEE 754 also specifies two classes of extended floating-point formats: single-extended and double-extended. The standard does not prescribe the exact precision and size of these formats, but it does specify the minimum precision and size. For example, an IEEE double extended format must have a significand precision of at least 64 bits and occupy at least 79 bits overall. IEEE 754 also specifies the precise layout of bits in a single and double precision.

The accuracy requirements of IEEE 754 on floating-point operations are: *add, subtract, multiply, divide, square root, remainder, round numbers in floatingpoint format to integer values, convert between different floating-point formats, convert between floating-point and integer formats, and compare.* Note that most mathematical functions that we use in scientific calculation are not included in the requirements here.

*A.3 Accuracy of transcendental functions*

As we summarized in the previous subsection, IEEE 754 specifies add, subtract, multiply, divide, square root, and some other arithmetic operations, but not transcendental functions such as sine or exponential. Hence, we have to trust what computer vendors say as to the accuracy of numerical transcendental functions that we use on the computer. Collection of what computer vendors officially say and/or publish about the accuracy of mathematical subroutines might come in handy when we check the accuracy of these functions. Here is a short list of URLs of webpages where several computer vendors and groups release relevant information to the public. On other platforms, you might as well be able to consult this kind of information as on-line manual pages, such as `math(3M)` on FreeBSD.

- Intel (IA–32) : *IA–32 Intel Architecture Software Developer's Manuals*
  `http://developer.intel.com/design/pentium4/manuals/index_new.htm#sdm_vol1`
  (See "8.3.10. Transcendental Instruction Accuracy")
- SGI (IRIX 6.5) : Online manual of `TRIG(3M)` of IRIX 6.5
  `http://techpubs.sgi.com/library/tpl/cgi-bin/getdoc.cgi?coll=0650&db=man&fname=/usr/share/catman/p`

_man/cat3/standard/fatan2.z&srch=sin

- SPARC : *Numerical Computational Guide* http://docs.sun.com/app/docs/doc/806-3568?q=numerical+computation+guide (See Chapter 3 “The Math Libraries”, especially the section “Implementation Features of libm and libsunmath”)
- Fujitsu (UXP/V on VPP5000) : Though there is no official document on the accuracy of numerical transcendental functions on this platform, the development team kindly measured the error of several transcendental functions in double precision arithmetic and provided us with the result. A part of the result that is relevant to this manuscript is: In C V20L20 (vector), maximum errors in sin() is 1.67, cos() is 1.77, and tan() is 2.23. In C V20L20 (scalar), maximum errors in sin() is 0.67, cos() is 0.89, and tan() is 1.38. The unit of error is ulp, and the tested argument range is $[-\pi/4, \pi/4]$.
- GNU libc : *The GNU C Library* http://www.gnu.org/software/libc/manual/html_node/Errors-in-Math-Functions.html (See Section “19.7 Known Maximum Errors in Math Functions”)

**References**


Danby, J.M.A. (1992) *Fundamentals of Celestial Mechanics (second edition, third printing)*, Willmann-Bell Inc., Richmond, Virginia.

Fukushima, T. (1997) A method solving Kepler's equation without transcendental function evaluations, *Celes. Mech. Dyn. Astron.*, 66, 309–319.

Green, R. (2003) Faster Math Functions, in *2003 GDC Written Proceedings*, Game Developers Conference, San Francisco, CA, http://www.gdconf.com/archives/2003/index.htm, http://www.research.scea.com/research/pdfs/RGREENfastermath_GDC02.pdf.

Ito, T. (1997) VPP300 series in National Astronomical Observatory, *FUJITSU Sci. Tech. J.*, 33, 74–87.

Ito, T. and Tanikawa, K. (2002) Long-term integrations and stability of planetary orbits in our solar system, *Mon. Not. R. Astron. Soc.*, 336, 483–500.

Kinoshita, H., Yoshida, H., and Nakai, H. (1991) Symplectic integrators and their application to dynamical astronomy, *Celes. Mech. Dyn. Astron.*, 50, 59–71.

Lichtenberg, A.J. and Lieberman, M.A. (1992) *Regular and Chaotic Dynamics*, Springer-Verlag, New York.

Standish, E.M. (1990) The observational basis for JPL's DE 200, the planetary ephemerides of the astronomical almanac, *Astron. Astrophys.*, 233, 252–271.

Wisdom, J. and Holman, M. (1991) Symplectic maps for the *N*-body problem, *Astron. J.*, 102, 1528–1538.

Yoshida, H. (1990) Conserved quantities of symplectic integrators for Hamiltonian systems, preprint.

Published by
National Astronomical Observatory of Japan
Mitaka, Tokyo

平成 17 年 10 月 25 日印刷
平成 17 年 10 月 31 日発行

発行者　国　立　天　文　台
〒181-8588 東京都三鷹市大沢2-21-1
印刷所　株 式 会 社　芳　文　社
〒194-0035 東京都町田市忠生1-18-18

(2005. 10. 1100)