ELSEVIER

Computer Physics Communications 120 (1999) 41–56

Computer Physics Communications

www.elsevier.nl/locate/cpc

Numerical computation of solar neutrino flux attenuated by the MSW mechanism

Jai Sam Kim [a], Yoon Sang Chae [a,1], Jung Dae Kim [b,2]

[a] *Department of Physics, Pohang University of Science and Technology, Pohang 790-784, South Korea*

[b] *Department of Physics, Yonsei University, Seoul 120-749, South Korea*

Received 4 January 1999

Abstract

We compute the survival probability of an electron neutrino in its flight through the solar core experiencing the Mikheyev–Smirnov–Wolfenstein effect with all three neutrino species considered. We adopted a hybrid method that uses an accurate approximation formula in the non-resonance region and numerical integration in the non-adiabatic resonance region. The key of our algorithm is to use the importance sampling method for sampling the neutrino creation energy and position and to find the optimum radii to start and stop numerical integration. We further developed a parallel algorithm for a message passing parallel computer. By using an idea of job token, we have developed a dynamical load balancing mechanism which is effective under any irregular load distributions. © 1999 Elsevier Science B.V. All rights reserved.

PACS: 02.70.–c; 14.60.Pq; 26.65.+t

1. Introduction

In recent years the utility of neutrinos in probing the interior structures of stars and their violent deaths has been increasingly recognized. 30 years ago Davis et al. [1] started their bold and pioneering experiment to measure neutrino fluxes radiating from the Sun as byproducts of nuclear fusion. The most striking event in neutrino astronomy occurred on 23 February 1987, when an exploding supernova was caught in action [2] and 24 neutrinos from that supernova were detected independently at several neutrino laboratories worldwide [3–5]. Neutrinos that were introduced to balance the momentum deficit in beta decays have undoubtedly become important particles composing the Universe. They carry away most of the explosion energy of type II supernova and some people [6] even proposed neutrino stars and galaxies in an attempt to provide missing mass of the Universe. Thus the advent of neutrino astronomy is very timely and a flurry of activities has been generated in this field [7].

[1] Currently at Hyundai Electronics Inc.; e-mail: jsk@vision.postech.ac.kr.

[2] Formerly at Korea Institute for Advanced Science.

PII S0010-4655(99)00218-0

On the theoretical side a few groups [8,9] have been perfecting their computer codes simulating physical processes taking place in the solar interior. However, the experimental data and the theoretical predictions do not agree so far, which has been termed the solar neutrino puzzle (for a review, see [10,11]). The discrepancies between the standard solar model (SSM) predictions and the measured capture rates are so large that a new mechanism was needed to reduce the solar neutrino flux. This could be accomplished if we abandon the idea of masslessness of the neutrino. Neutrinos can be given masses in various ways. The see-saw mechanism and its extensions [12–14] are considered to be the most plausible ideas.

If neutrinos acquire masses then the flavor eigenstates are no longer energy eigenstates. The amplitudes in the flavor eigenstate basis are time-dependent quantities and thus neutrino flavors change periodically [15]. This implies that a neutrino born as an electron neutrino in the solar core can transmute into a muon neutrino or a tau neutrino on its way to the Earth. There have been several experiments to measure oscillations of neutrinos produced in accelerator laboratories and those produced in the upper atmosphere as decay products from cosmic rays. The Super-Kamiokande experiment [16] for atmospheric neutrinos recently set a stringent limit to the neutrino mass differences. It has established that neutrinos are no longer massless.

Neutrinos interact with electrons in matter and acquire effective masses. Mikheyev–Smirnov–Wolfenstein [17,18] proposed a mechanism that the flavor transmutation can be enhanced in a medium. The transition rates between mass eigenstates can be obtained approximately using the Landau–Zener formula [19,20]. There have been many attempts to obtain the rates approximately in analytic forms [21–24]. However, all these efforts are for two generation neutrinos only and of limited usefulness. It would be most reliable and useful if we could obtain them numerically for three generation neutrinos. However, the numerical computation requires excessive amount of number crunching and a new generation supercomputer was awaited.

In the past 10 years supercomputer technologies such as RISC processors, both hardware and software technologies for interprocessor communication, memory technologies, operating systems for multicomputer systems, etc., advanced significantly. Especially technologies for parallel computer architectures [25–27] matured so much that the solar neutrino flux can nowadays be computed without too much burden within a tolerable time span.

We present a straightforward numerical algorithm that computes the survival probabilities of the solar neutrinos. Since it is extremely time-consuming we have developed a parallel algorithm that can be used in message passing parallel computing machines. In Section 2 we briefly review solar neutrino physics. In Section 3 we set up equations to solve. In Section 4 we explain our sequential algorithm and then in Section 5 we present our parallel algorithm and test results.

2. Overview

2.1. Neutrino oscillations

The standard $SU(2)_L \times U(1)_Y$ gauge theory for the electroweak interaction with massless neutrinos has become shaky by the recent SuperKamiokande data for atmospheric neutrinos. Neutrino oscillation between flavor eigenstates necessarily implies the existence massive neutrinos. It can be explained using a simple quantum mechanical argument. Let us consider two generations of neutrino for simplicity. A neutrino state in the flavor (or weak) eigenstates can be expressed as a superposition of the mass eigenstates as follows:

$$\begin{pmatrix} \nu_e \\ \nu_\mu \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \nu_1 \\ \nu_2 \end{pmatrix}, \tag{1}$$

where θ is the vacuum mixing angle. If we assume that the neutrinos have non-degenerate masses, then the equation of motion for neutrino states in vacuum can be written as follows:

$$i\frac{d}{dt}\begin{pmatrix}\nu_1\\ \nu_2\end{pmatrix} = \frac{1}{2E}\begin{pmatrix}2E^2+m_1^2 & 0\\ 0 & 2E^2+m_2^2\end{pmatrix}\begin{pmatrix}\nu_1\\ \nu_2\end{pmatrix}. \tag{2}$$

The common diagonal elements $2E^2$ give rise to an overall phase and thus can be subtracted harmlessly. Substituting the transformation equation (1) and subtracting a term, $(m_1^2\cos^2\theta + m_2^2\sin^2\theta) - (1/2)\Delta_{21}\cos(2\theta)$, from the diagonal elements, Eq. (2) can be rewritten as

$$i\frac{d}{dt}\begin{pmatrix}\nu_e\\ \nu_\mu\end{pmatrix} = \frac{1}{4E}\begin{pmatrix}-\Delta_{21}\cos(2\theta) & \Delta_{21}\sin(2\theta)\\ \Delta_{21}\sin(2\theta) & +\Delta_{21}\cos(2\theta)\end{pmatrix}\begin{pmatrix}\nu_e\\ \nu_\mu\end{pmatrix}, \tag{3}$$

where $\Delta_{21} = m_2^2 - m_1^2$.

Neutrino productions and detections take place in its flavor states. We can obtain the survival probability for an electron neutrino in vacuum as follows:

$$P(\nu_e \to \nu_e; t) = |\langle\nu_e(t)|\nu_e(0)\rangle|^2 = 1 - \sin^2(2\theta)\sin^2\left(\frac{\Delta_{21}}{4E}t\right). \tag{4}$$

Non-degenerate neutrino masses and a nonzero mixing angle give rise to the change of neutrino flavor as it evolves in space or time.

2.2. MSW effect

When a neutrino propagates in matter, it weakly interacts with the particles in the medium. We assume the coherent interactions with matter so that the medium remains unchanged and it allows the interference of scattered and unscattered neutrino wave functions. Coherent elastic scattering gives rise to the effective potential for neutrino due to weak interactions with the particles in the medium. The detailed calculations to derive the effective potential from the low-energy weak interaction Hamiltonian can be found in Refs. [11,28]. With the additional potential, the equation of motion in matter is modified as follows:

$$i\frac{d}{dt}\begin{pmatrix}\nu_e\\ \nu_\mu\end{pmatrix} = \frac{1}{4E}\begin{pmatrix}-\Delta_{21}\cos(2\theta)+A_c & \Delta_{21}\sin(2\theta)\\ \Delta_{21}\sin(2\theta) & \Delta_{21}\cos(2\theta)\end{pmatrix}\begin{pmatrix}\nu_e\\ \nu_\mu\end{pmatrix}, \tag{5}$$

where we subtracted a mass squared term A_n induced by the weak neutral current interaction, which are commonly present in the diagonal elements. The mass squared term A_c induced by the weak charged current interaction appears only in the (1–1) position and is given by

$$A_c = 2\sqrt{2}\,G_F E N_e\,, \tag{6}$$

where G_F is the Fermi coupling constant, E the neutrino energy and N_e the electron number density in the medium.

When the variation of the induced mass is very small over the oscillation length $L_{\rm osc} = 4\pi E/\Delta_{21}$, we can introduce the effective masses and the effective mixing angle ϕ as follows:

$$\mu_1^2 = \frac{\Sigma}{2} + \frac{A_c}{2} - \tfrac{1}{2}[(\Delta_{21}\cos(2\theta) - A_c)^2 + (\Delta_{21}\sin(2\theta))^2]^{1/2}, \tag{7}$$

$$\mu_2^2 = \frac{\Sigma}{2} + \frac{A_c}{2} + \tfrac{1}{2}[(\Delta_{21}\cos(2\theta) - A_c)^2 + (\Delta_{21}\sin(2\theta))^2]^{1/2}, \tag{8}$$

where $\Sigma \equiv m_1^2 + m_2^2 + A_n$ and

$$\tan(2\phi) = \frac{\tan(2\theta)}{1 - A_c/\Delta_{21}\cos(2\theta)}. \tag{9}$$

The survival probability for an electron neutrino which is produced in matter at time $t = 0$ and detected in vacuum at time t is given by

$$P(\nu_e \to \nu_e; t) = \cos^2\phi\cos^2\theta + \sin^2\phi\sin^2\theta + \tfrac{1}{2}\sin(2\theta)\sin(2\phi)\cos\left(\frac{\delta(t)}{2E}\right), \tag{10}$$

where $\delta(t) = \int_0^t [\mu_2^2(t') - \mu_1^2(t')]\, dt'$.

Mikheyev–Smirnov–Wolfenstein [17,18] noticed that the matter effects may change the behavior of the survival probability drastically if the neutrino passes the resonance point ($\phi = \pi/4$). When the medium density rapidly varies, we can use the Landau–Zener level crossing probability P_{LZ} to obtain the asymptotic survival probability as follows:

$$P(\nu_e \to \nu_e) = \tfrac{1}{2}[1 + (1 - 2P_{\mathrm{LZ}})\cos(2\theta)\cos(2\phi)], \tag{11}$$

where $P_{\mathrm{LZ}} = \exp(-\frac{1}{4}\pi\gamma)$ and

$$\gamma = \frac{\Delta_{21}\sin^2(2\theta)}{E|\dot{A}/A|_R}. \tag{12}$$

The dot over A in the denominator of γ means time derivative and the subscript R stands for the calculation at the resonance point. The detailed formulae for the MSW effect with three generation neutrinos will be given in the next section.

2.3. *Landau–Zener approximations vs numerical computation*

There have been many attempts to compute the neutrino survival probability. With the assumption of a linear or exponential electron density profile, analytic formulae for the Landau–Zener transition probability were obtained in the two generation case [21–24]. In this case even exact solutions of the neutrino evolution equations were obtained [29,30]. Direct numerical computation was done also for the two generation case [31,32].

For the three generation case results from the two generation case were extended with some limitations [28,33–45]. In all these works that used the Landau–Zener formula, the possibility that the most neutrino productive region and the non-adiabatic resonance region overlap was overlooked. There is no approximate formula like the Landau–Zener formula to use in this case. Another difficulty arises when two or three non-adiabatic regions overlap.

Thus an efficient numerical algorithm is needed to cover all possible cases. To the best of our knowledge, there has been no publication on direct numerical computation for the three generation case. The neutrino evolution equations are plain ordinary differential equations and do not require fancy and sophisticated numerical techniques. However, the amount of computing is excessive and a new generation supercomputer is needed. In this paper we shall present an algorithm for a massage passing parallel computer.

3. Detailed formalism

3.1. *Wave equations in weak and mass eigenstate basis*

We now set up equations needed for tracing the flight of a neutrino through the solar core interacting with matter via the neutral and charged currents and thus experiencing the MSW mechanism. In the weak eigenstate basis let us write the neutrino wavefunction $\psi(t)$ as

$$\psi(t) = \sum_{\alpha=e,\mu,\tau} a_\alpha(t)|\nu_\alpha\rangle. \tag{13}$$

Then its wave equation is written as

$$i\frac{d}{dt}\begin{pmatrix} a_e(t) \\ a_\mu(t) \\ a_\tau(t) \end{pmatrix} = \frac{1}{2\beta}\begin{pmatrix} M_{11}+A_c(t) & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix}\begin{pmatrix} a_e(t) \\ a_\mu(t) \\ a_\tau(t) \end{pmatrix}, \tag{14}$$

where $\beta = E/R_\odot(\mathrm{MeV/cm})$ and $t = R/R_\odot$.

The mass matrix is nondiagonal in the weak eigenstate basis,

$$M_{ij} = \sum_k U_{ik}^\dagger m_k^2 U_{kj}\,, \tag{15}$$

where the unitary matrix U transforms the mass eigenstate into the weak eigenstates,

$$\nu_\alpha = \sum_{i=1}^{3} U_{\alpha i}\nu_i\,, \quad \alpha = e,\mu,\tau\,. \tag{16}$$

The Particle Data Group adopts the convention for the mixing matrix,

$$U = \begin{pmatrix} C_1C_3 & S_1C_3 & S_3 \\ -S_1C_2 - C_1S_3S_2 & C_1C_2 - S_1S_3S_2 & C_3S_2 \\ S_1S_2 - C_1S_3C_2 & -C_1S_2 - S_1S_3C_2 & C_3C_2 \end{pmatrix}, \tag{17}$$

where $C_i \equiv \cos\theta_i$ and $S_i \equiv \sin\theta_i$ and the three mixing angles θ_1, θ_2, and θ_3 roughly measure mixing between mass eigenstates (1–2), (2–3), and (1–3), respectively. We have neglected the CP violating phase, which is irrelevant in this problem.

In the most general situation, one should integrate Eq. (14) numerically, which seems to be a trivial job. However, for small values of Δ_{12}/E and Δ_{23}/E the mass matrix elements become large and one should take a special care about the errors.

If we change the basis to the mass eigenstate the wave equation becomes

$$i\frac{d}{dt}\begin{pmatrix} a_1(t) \\ a_2(t) \\ a_3(t) \end{pmatrix} = \left[\frac{1}{2\beta}\begin{pmatrix} \mu_1^2(t) & 0 & 0 \\ 0 & \mu_2^2(t) & 0 \\ 0 & 0 & \mu_3^2(t) \end{pmatrix} - i\begin{pmatrix} 0 & \Phi_{21}(t) & \Phi_{31}(t) \\ -\Phi_{21}(t) & 0 & \Phi_{32}(t) \\ -\Phi_{31}(t) & -\Phi_{32}(t) & 0 \end{pmatrix}\right]\begin{pmatrix} a_1(t) \\ a_2(t) \\ a_3(t) \end{pmatrix}. \tag{18}$$

The off-diagonal elements Φ_{ij} are given by

$$\begin{aligned} \Phi_{12}(t) &= \dot{\phi}_1 + s_3\dot{\phi}_2\,, \\ \Phi_{13}(t) &= -s_1c_3\dot{\phi}_2 + c_1\dot{\phi}_3\,, \\ \Phi_{23}(t) &= c_1c_3\dot{\phi}_2 + s_1\dot{\phi}_3\,, \end{aligned} \tag{19}$$

where ϕ_i are mixing angles in matter and $c_i \equiv \cos\phi_i$ and $s_i \equiv \sin\phi_i$. Since $\dot{\phi}_i$ are proportional to the spacial derivative of the electron density $\dot{\rho}$, all Φ_{ij} are proportional to $\dot{\rho}$. The off-diagonal terms come about because the mass eigenstates themselves depend on the electron density and are varying as the neutrino passes through the solar core.

The derivatives of the angles are with respect to time or equivalently neutrino's path length $r \equiv ct$. The mixing angles ϕ_i in matter can be obtained from the mixing matrix which diagonalizes the mass matrix. Define a pair of parameters, $g_1 \equiv U_{23}/U_{13} = c_3s_2/s_3$ and $g_2 \equiv U_{33}/U_{13} = c_3c_2/s_3$. Then we obtain

$$\tan\phi_3 = 1/\sqrt{g_1^2 + g_2^2}\,,$$
$$\tan\phi_2 = |g_2|/\sqrt{g_1^2 + g_2^2}\,. \tag{20}$$

It takes some more algebra to obtain an expression for ϕ_1. Define $g_3 \equiv U_{21}/U_{11} = -[(c_2/c_3)\tan\phi_1 + (s_3 s_2/c_3)]$. Then we can derive

$$\tan\phi_1 = -[(s_3 s_2/c_3) + g_3](c_3/c_2)\,. \tag{20'}$$

In order to integrate the six component ordinary differential equations (18) in the mass eigenstate basis, one has to first diagonalize the mass matrix to get $\mu_i^2(t)$ at each time step. One then has to find the mixing angles in matter and then compute their time derivatives. Thus it is not a good idea to integrate the wave equation in the mass eigenstate basis in general.

3.2. Adiabaticity

In a region where the off-diagonal terms are negligibly small the mass-based equation (18) is more advantageous. In this case the equation becomes practically diagonal and each mass eigenstate evolves independently without influencing each other. Such a region is called the adiabatic region. Instead of integrating the wave equation one obtains the solution simply by performing the phase integrals,

$$a_i(t) = a_i(t_0)\exp\left(-i\int_{t_0}^{t}\frac{\mu_i^2(t')}{2\beta}\,dt'\right). \tag{22}$$

One still has to diagonalize the mass matrix to carry out the integration but it is much easier than solving Eq. (18).

Let us define a set of adiabaticity parameters,

$$\gamma_{ij} \equiv \frac{(\mu_j^2 - \mu_i^2)}{|2\beta\Phi_{ji}|}\,, \tag{23}$$

which reduces to Eq. (12) in the two generation case. When all parameters are much larger than unity,

$$\gamma_{ij} \gg 1\,, \tag{24}$$

the off-diagonal terms can be neglected and there will be no transitions between mass eigenstates. The adiabaticity conditions (23) are somewhat complicated in the case of three neutrinos. They depend on the initial parameters, $(E, \Delta_{ij}, \theta_i)$ in a complicated way. For example, in the two generation case, the adiabaticity condition (23) can be written as

$$\frac{\Delta}{E}\frac{\sin^2(2\theta)}{\cos(2\theta)} \gg 3\times 10^{-9}\,, \tag{25}$$

where Δ is in eV^2 and E is in MeV. In Eq. (24) we have used the fact that $|\dot{N}/N|$ is approximately constant for $r \geq 0.2R_\odot$ and $|\dot{N}/N|_R \simeq 3\times 10^{-15}$ eV. Thus when the neutrino energy is high or the mass square differences are very small or θ is small, one begins to see non-adiabatic resonance regions. In the three generation case, the situation is more complicated but these trends are qualitatively identical.

If we assume the mass hierarchy $\mu_1 < \mu_2 < \mu_3$, then the (1–3) mixing is always small. Thus practically we only need to worry about the (1–2) and (2–3) mixings. The Landau–Zener formula for the transition probability between two mass eigenstates, i and j, is given by

$$P_{\mathrm{LZ}}(i \to j) = \exp\left(-\frac{\pi}{4}\gamma_{ij}\right). \tag{26}$$

In non-adiabatic resonance regions the elements of the mass matrix become large and it is wise to have the weak eigenstates evolve according to Eq. (14). However, integrating the wave equations in any basis is very time-consuming because one component varies rapidly and holds the others down due to the coupling terms.

3.3. Evolution through the non-resonance region

Let us denote the mixing matrix at radius r_H as $U(\theta_i)$ where θ_i are the three mixing angles at r_H and the mixing matrix in matter at a radial position r_L as $U(\phi_i)$ where ϕ_i represents the three effective mixing angles at r_L. When a neutrino in a mixed state passes through the non-resonance region from r_L to r_H, then to a very good approximation, we can express its transition probability into a weak eigenstate σ as

$$P(\nu_{\mathrm{mix}} \to \nu_\sigma) = \left|\nu_\sigma^\dagger U(\theta) \exp\left(-i \int_{r_L}^{r_H} \frac{\mu_i^2(t')}{2\beta}\, dt'\right) U^\dagger(\phi)\nu_{\mathrm{mix}}\right|^2. \tag{27}$$

If we take an average of the transition probability over paths, then the exponential factors drop out and we get

$$\langle P(\nu_{\mathrm{mix}} \to \nu_\sigma)\rangle = \sum_{i=1}^{3} [\,|U_{\sigma i}(\theta)|^2 |\sum_\rho U_{\rho i}^*(\phi) a_\rho|^2\,]. \tag{28}$$

For a real mixing matrix U which is the case when the CP violating phase in the mixing matrix is zero, it is reduced to

$$\langle P(\nu_{\mathrm{mix}} \to \nu_\sigma)\rangle = \sum_{i=1}^{3} [\,(U_{\sigma i}(\theta))^2 |\sum_\rho U_{\rho i}(\phi) a_\rho|^2\,]. \tag{29}$$

4. Sequential algorithm

We now explain our computation procedure.

4.1. Sampling neutrino energy and creation position

We sample the neutrino energy according to the energy spectrum $f_i(E)$ predicted by the standard solar model and the capture rates $x_a(E)$ of detectors. We use the data provided by Bahcall et al. [8,46]. What we use for the sample function is the product $\mathcal{F}(E) \equiv f_i(E) \times x_a(E)$. For example, for the boron neutrinos captured by the chlorine and gallium detectors, we made a plot of $\mathcal{F}(E)$ in Fig. 1. We adopt the importance sampling method frequently used in Monte Carlo methods [47–49]. We pick more sample points in an interval where the sample function has larger values. Our choice of the sampling points according to $\mathcal{F}(E)$ is listed in Table 1. We strategically placed sample points so that all kinds of neutrinos that contribute to the chlorine detector are represented according to their importance.

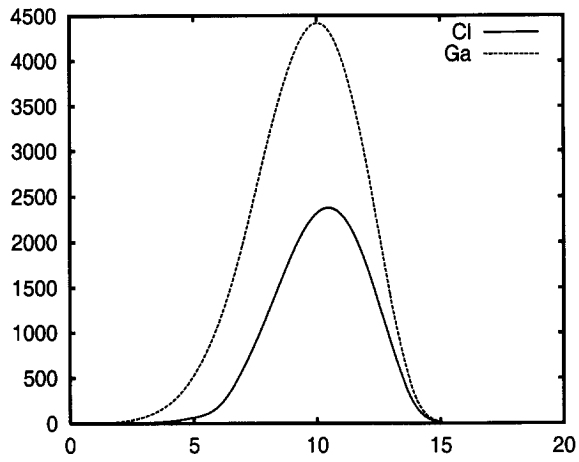

Fig. 1.

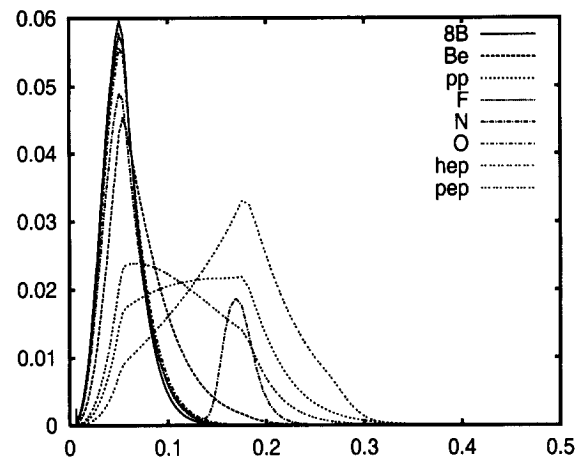

Fig. 2.

Fig. 1. Boron neutrino energy sampling functions: $\mathcal{F}_a(E) = f_B(E) \times x_a(E)$ with E in MeV.

Fig. 2. Neutrino radius sampling function: $r_i(t)$ with $t = R/R_\odot$.

Table 1
Sampling points of neutrino energy focused on B neutrino

Interval	Number	Increment	Comment
0.737–1.862	10	0.125	N, O, Be, pep
2.317–8.004	12	0.517	B
8.155–12.65	30	0.155	B
12.65–14.65	4	0.500	B
14.65–18.65	4	1.000	hep

Table 2
Sampling points of neutrino creation position focused on B neutrino

Interval	Number	Increment	Comment
0.004–0.020	5	0.0040	
0.020–0.100	32	0.0025	bodies of B, Be, O, N, pep; head of hep
0.100–0.128	8	0.0035	tails of B, Be; bodies of pep, hep
0.128–0.253	10	0.0125	tail of Be; bodies of N, pep, hep
0.253–0.353	5	0.0200	tails of pep, hep

We then sample neutrino creation positions according to the radial spectrum $r_i(t)$ $(t \equiv R/R_\odot)$ predicted by the standard solar model. Here i stands for neutrino sources like pp, B, Be, etc. We use the Bahcall data again. In Fig. 2, the radial spectra of neutrinos from various sources are depicted. The spectra for the boron, fluorine, and oxygen neutrinos are very similar and we can use the same sampling table. We again use the importance sampling method. Our choice of the sampling points is listed in Table 2. Our basic sampling strategy is the same as before.

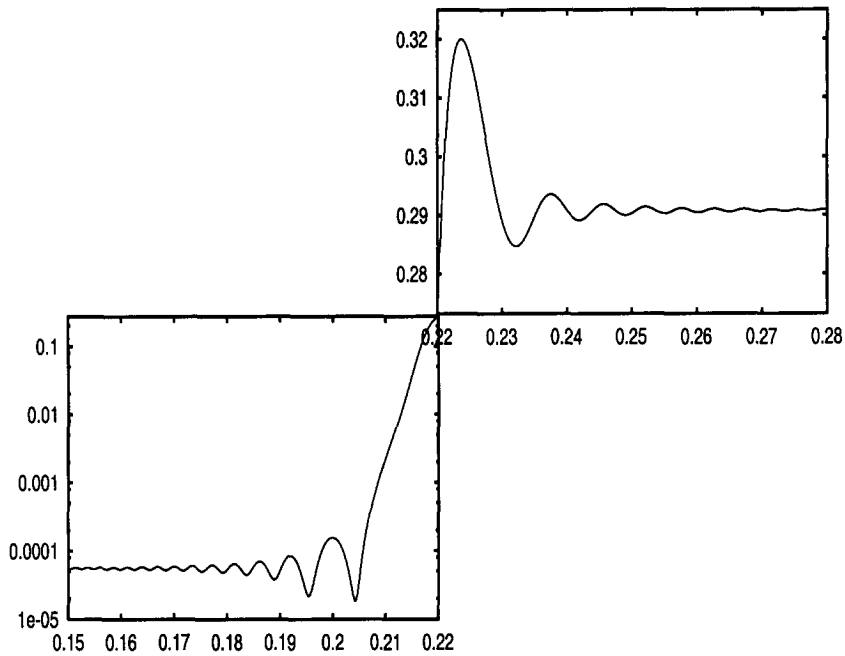

Fig. 3. A typical neutrino oscillation with $\Delta_{12} = 10^{-5}$ eV^2, $\sin^2(2\theta_1) = 0.001$, $\Delta_{23} = 2.2 \times 10^{-3}$ eV^2, $\sin^2(2\theta_3) = 0.9973$, $E_\nu = 5$ MeV. The limits of the non-adiabatic region are BAND(1) = 0.2092, BAND(2) = 0.2262 with $\gamma_c = 17.59045$. The abscissa refers to a_1^2 with different scales and the ordinate refers to $t = R/R_\odot$.

4.2. Locating the non-adiabatic regions

Before we numerically integrate the neutrino evolution equations, we check whether and where the non-adiabatic regions are located by computing the adiabaticity parameters, γ_{ij}. Using an elementary root finding algorithm, we check where $\gamma_{ij}(t_c) = \gamma_c$, with $\gamma_c = 17.59045$, for example. This value of γ_c yields $P_{\rm LZ} = 10^{-6}$. It takes a little sophisticated algorithm to find out the beginning and end points of each non-adiabatic region within the interval $0 < t < 1$. Their existence requires large values of E/Δ_{ij} or small mixing angles. The non-adiabatic regions may overlap in a complicated way. We shall denote the smallest of the beginning points, BAND(1) and the largest of the end points, BAND(2).

4.3. Evolution from birth to the edge of non-adiabatic region

Suppose that the neutrino is born at a radial position t in a non-resonance region. Then as it moves out of the solar core until it enters into the non-adiabatic resonance region, its amplitudes in mass eigenstate remain unchanged. Thus we do not have to integrate the evolution equation. Instead, we simply decompose the electron neutrino eigenstate into mass eigenstates at the birth place and then reassemble them into the weak eigenstates at the edge of the non-adiabatic regions, BAND(1). Thus we transport the neutrino adiabatically.

Our computer experiment indicates that the amplitudes begin to oscillate before the neutrino reaches the edge (see Fig. 3). Thus we have to start numerical integration well before the edge to keep the required accuracy where the amplitude of oscillating amplitude is smaller than the tolerance. The optimum starting location $t_L \leq$ BAND(1) depends on the particular set of parameters and its determination is the most difficult part of our algorithm. We have to estimate it without actual numerical integration.

We measured BAND(1) $- t_L$ relative to the width of the non-adiabatic region, $\delta \equiv$ BAND(2) $-$ BAND(1), with

$\gamma_c = 17.59045$ and a tolerance of 5×10^{-5}. For various combinations of $(\Delta_{12}, \theta_1, E_\nu)$ with $\Delta_{23} = 2.2 \times 10^{-3}$ eV^2 and $\theta_2 = \theta_3 = 43.5°$, we observed $(\texttt{BAND(1)} - t_L)/\delta < 2$. Thus we set $t_L = \texttt{BAND(1)} - 2\delta$.

When a neutrino is born at $t < t_L$ it is transported adiabatically until t_L and then its evolution equation is numerically integrated. In other cases the integration starts at the neutrino birth position. In this way we can save a lot of computing time without loosing accuracy. It is very effective when the non-adiabatic region lies much away from the solar center. This is the case when Δ_{12}/E_ν or θ_1 are small.

4.4. Integration through the non-adiabatic resonance region

Now a neutrino is created at a radial position t with energy E, chosen from Tables 1 and 2. Then we let it start its journey towards the Earth. Once it passes beyond t_L or if the neutrino is born within the non-adiabatic region, we integrate the six component ordinary differential equation in the weak eigenstate basis, using the standard ODE solver such as DDEQBS of the CERN library [50] with tolerance $\sim 10^{-5}$. Higher accuracy is possible but unnecessary due to uncertainties in the solar model predictions and detector sensitivities. The routine DDEQBS is an implementation of the Bulirsch–Stoer algorithm [51], which is known to be one of the fastest algorithms. It is carried out from t_L or the creation position till the upper limit, BAND(2). Due to rapid oscillations of $a_\alpha(t)$, very fine time steps are needed to keep the errors within tolerance. The subroutine DDEQBS survived this stringent test with modest efficiency.

Beyond the upper limit, namely entering into the non-resonance region, the mass eigenstate components of the neutrino oscillate about almost constant average values. The neutrino oscillation lengths, $L_{\rm osc} = 4\pi E_\nu/\Delta_{ij}$ are computed at $t = \texttt{BAND(2)}$ and $t \sim 0.99$. Then the increment dt is fixed at 1/720th of the smallest of the oscillation lengths. The amplitude of the mass eigenstate neutrino, $p_1 = |a_1|^2$, is monitored at every time step. We need to diagonalize the mass matrix to get mass eigenstate amplitudes but this overhead is much less than that of blind integration all the way to $t = 1$. We could monitor $p_e = |a_e|^2$ instead. Due to beats occurring between mass eigenstates with different oscillation frequencies, it is difficult to recognize where the electron neutrino wave stabilizes.

We thus monitor at least 720 times as the neutrino oscillates one cycle. We especially monitor whether the amplitude passes extrema. After observing some number (about 10) of extrema, we begin to take the average of maxima and minima. We then check if the average value has converged to a constant value within a tolerance, $\sim 5 \times 10^{-5}$. Once we confirm the convergence we enter into the next phase of integration. We save the value of t as t_H and all six neutrino components, $a_\alpha(t)$.

4.5. Integration beyond the non-adiabatic region

There is no need for direct numerical integration in this region. For transporting a neutrino in the non-resonance region we use the accurate and efficient formula, Eq. (28), with $t = t_H$ as the beginning point and $t = 1$ as the end point. It is worrisome that different choices of γ_c may yield a different set of BAND(I) and result in different values for the survival probabilities. We have tried different values for γ_c and obtained the same result within the tolerance. Thus our results are very reliable.

In this way we save a big chunk of computing time than numerically integrating the evolution equations from the neutrino birth point to the solar surface. The key of our algorithm is to find the optimum radii to start and stop numerical integration and then to use the accurate and quick adiabatic formula. In order to help the reader to understand our algorithm, we made a plot of a neutrino oscillation in a typical setting in Fig. 3.

When there are well separated multiple resonance regions we can apply the above algorithm to pass each non-adiabatic resonance region. However, if the regions are closely adjacent to each other or overlap it is better to merge them into a single region.

The idea of using analytic formula in the non-resonance region and numerically integrating in the non-adiabatic region was advocated by Messiah [52]. This hybrid method was exploited for the two generation

Table 3
Computation times for jobs in seconds; the number in parentheses is the occurrence number of non-adiabatic regions in a job

$\sin^2(2\theta_1)$	Δ_{12}		
	10^{-3}	10^{-4}	10^{-5}
1×10^{-4}	6(0)	147(10)	5895(60)
1×10^{-3}	6(0)	189(7)	8972(60)
1×10^{-2}	6(0)	6(0)	9185(56)
1×10^{-1}	6(0)	6(0)	6(0)

case in [31].

5. Parallel algorithm

5.1. Grain size

In order to assess the parameter values that agree with the experimental data, we need to numerically solve the six component ordinary differential equations, from various neutrino creation positions to the solar surface and at various neutrino energies, for a given set of parameters, $(\Delta_{12}, \Delta_{23}, \theta_1, \theta_2, \theta_3)$. θ_2 has no effect on the survival probability. We set $\sin^2(2\theta_2) = 0.997$ in accordance with the recent Super-Kamiokande data for the atmospheric neutrinos [16]. So we need to sweep over six parameters, $(\Delta_{23}, \theta_3, \Delta_{12}, \theta_1, E, t)$. While holding (Δ_{23}, θ_3) fixed we sweep over $(\Delta_{12}, \theta_1, E, t)$. Thus surveying the entire parameter space requires a tremendous amount of computation. Since each neutrino is traced independently, a workstation cluster or a medium-grained massively parallel computer is ideal for the task.

On a message passing parallel computer [25], it is convenient to set up a virtual machine complex where there is a host node and many worker nodes. We choose an operation mode that the host node assigns jobs for worker nodes, collects computed results, then writes the combined results to a file. A worker node computes the survival probabilities of electron neutrinos born at 60 selected radial positions with 60 selected energies for a given set of (Δ_{12}, θ_1) and then sends the results to the host.

In Table 3, we give a sample of measured computing times with various combinations of (Δ_{12}, θ_1). In our test run, we chose a set, $(\Delta_{23} = 2.2 \times 10^{-3}\ \text{eV}^2, \sin^2(2\theta_3) = 0.997)$. We see that the computation time depends on the existence of the non-adiabatic regions and their widths crucially. It increases drastically at smaller values of Δ_{12} or θ_1. Using this information we need to choose the optimum grain size.

A mini-job is defined as the whole computation of 60 survival probabilities for a set of $(\Delta_{12}, \theta_1, E_\nu)$. A job is defined as the whole computation of 60×60 survival probabilities for a set of (Δ_{12}, θ_1). Thus a job consists of 60 mini-jobs. A task is defined as the whole set of jobs for a set of (Δ_{23}, θ_3). Thus it would be wise to nest the loops, E_ν-wise first (inner most loop) and θ_1-wise next (inner loop) and Δ_{12}-wise last (outer most loop).

We can now classify the grain sizes of the problem. (1) Coarse grain: the θ_1-loop is distributed. (2) Medium grain: the energy loop is distributed. (3) Fine grain: the radius loop is distributed.

As we can see from Table 3, the work load will be severely unbalanced if we choose the coarse grain. On the other hand, if we choose the fine grain the processors will be too frequently interrupted for communications. Thus the optimum grain size is the medium grain in this problem.

5.2. Job scheduling and data communication

There are a variety of communication routines provided in the MPI (Message Passing Interface) environment [53]. Since all communications take place only between the host and worker nodes but not among the

nodes in our problem, we use the simplest of them, MPI_Send and MPI_Recv. These are synchronous blocking routines. Asynchronous nonblocking routines are also available but they are more useful for an inhomogeneous problem requiring sophisticated communications among worker nodes (see, e.g., [54,55]). These routines label a message with a tag. If one calls MPI_Send to send a message with a given tag then one should call MPI_Recv with the same tag to receive that message.

The host and all nodes construct a unique job table in such a way that the mini-job number count is increased in the inner most E_ν loop and a set of three loop counts, the E_ν-loop count and the θ_1-loop count and the Δ_{12}-loop count corresponding to that mini-job number are stored in the table.

After performing initial household chores, each worker node sends to the host its node number with a READY tag. When the host in listening mode receives a worker node number with a READY tag, it replies to the sender by sending the mini-job number N with a JOB tag.

When a worker node receives a mini-job number with a JOB tag, it looks for the set of values $(\Delta_{12}, \theta_1, E_\nu)$ corresponding to that mini-job number from the precomputed table. At a given neutrino energy value of the 60 preset energy values (cf. Table 1) the node computes the boundaries of the non-adiabatic regions, BAND(1) and BAND(2). Then the worker node computes survival probabilities starting from 60 selected radial positions (cf. Table 2). After it sweeps through the radius loop it sends the mini-job number and the 60 survival probabilities back to the host with a RESULT tag.

Then the job hungry worker node sends to the host its node number with a READY tag again. The host in listening mode replies to the sender by sending the mini-job number from the queue. Thus the host and worker nodes cooperate to schedule jobs interactively for optimum load balance. The mini-job number is like a work token that the host gives to each worker node.

The host receives the mini-job number N and the computed results from worker nodes in a random order. It stores the survival probabilities at 60 consecutive memory locations starting from the $60\times$Nth position in the array. Since a mini-job number corresponds to a unique set of loop counts for $(\Delta_{12}, \theta_1, E_\nu)$, the host can retrieve the information from its own table. After all the computed data are received the host writes the ordered data to a file.

5.3. Test run on HP exemplar

We have used a 32 node HP X-Class Exemplar machine. Its architecture is a hybrid of a symmetric multiprocessor (SMP) design and a cache coherent non-uniform memory architecture (cc-NUMA). A node consists of a 200 MHz PA-8200 yielding 0.80 Gflops peak performance, a 2 MB D cache and 2 MB I cache. A processor agent controller (PAC) houses a pair of nodes. 8 PACs are connected to 8 memory access controls (MAC) via a nonblocking 8×8 crossbar switch with a bandwidth of 960 MB/sec per port. One MAC controls 4 banks of memory. A hypernode consists of 8 PACs, one crossbar switch, 8 MACs, 8 memory modules and 8 toroidal accesses (TAC). 8 MACs altogether connect 16 nodes to the 32-way interleaved physical memory of a capacity 256×16 MB. The inter-hypernode communication is controlled by TACs and its connection topology is a 2D mesh. All memory related buses and ports have bandwidths of 960 MB/sec and thus the machine is well balanced. Memory modules are not attached to individual nodes and thus a process can have much larger memory space than the capacity of a module.

The HP-UX operating system manages multiple processes for each node. A programmer can view the machine as a shared memory machine or a distributed memory machine. The HP-UX compiler allows one to parallelize loops or tasks by inserting a few compiler directives as in Cray. The SMP design is good for exploiting fine-grain parallelism. Distributed memory programming aided by the message passing interface allows more programming flexibility. The MPI library is provided with performance monitoring utilities. We used the primitive routines, MPI_Recv and MPI_Send.

We chose a 31×21 grid for (Δ_{12}, θ_1), logarithmically spaced. Δ_{12} (in eV^2) ranges between 10^{-6} and 10^{-3}. $\sin^2(2\theta_1)$ ranges between 10^{-4} and 1. In each run we executed one loop count JDL of Δ_{12} which is equivalent

Table 4
Total computation times for some time-consuming jobs; jobs with JDL > 21 do not take much time and thus not included, times are in seconds

JDL	Time	JDL	Time	JDL	Time	JDL	Time
1	549818	6	293690	11	113039	16	40757
2	490642	7	252483	12	90696	17	32026
3	437505	8	213698	13	74395	18	23056
4	385829	9	175545	14	60998	19	13516
5	338798	10	140963	15	50702	20	5477

Table 5
Computation times for a mini-task consisting of 8 jobs; N_p is the number of worker nodes, times are in seconds

N_p	1	2	3	4	5	6	7	8	9	10	11	12	total
12	785	747	754	752	769	745	751	756	774	780	767	762	9142
10	945	923	908	924	899	919	896	917	906	899			9136
8	1171	1128	1129	1148	1158	1128	1145	1136					9143
6	1554	1510	1517	1509	1528	1524							9141
4	2300	2275	2280	2288									9143

to 21 jobs or 1260 mini-jobs. Depending on the existence and widths of non-adiabatic regions the computation times vary significantly. For large values of Δ_{12} a run takes less than two hours but for small values it takes several days on a single CPU (see Table 4). The communication times are negligible in this number crunching problem. Thus the load balancing is the key issue in this problem.

In Table 5, we give a sample of computing times for runs with several different worker node numbers, N_p. We chose parametric values, $\Delta_{23} = 2.2 \times 10^{-3}$ eV^2, $\sin^2(2\theta_3) = 0.997$, $\Delta_{12} = 5 \times 10^{-5}$, and $\sin^2(2\theta_1)$ over the range, $\{1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2}, 5 \times 10^{-2}, 1 \times 10^{-1}, 5 \times 10^{-1}\}$. We see that the work load is pretty much balanced.

The HP-UX system manages job queues for multiple processes and some unlucky worker nodes may have a less share of CPU time than others. When the system is full of jobs, the operating system may even assign a few worker nodes to a single physical node. However, the HP-UX process management system gave almost equal share of CPUs for all nodes. Since the host distributes mini-jobs to available worker nodes, the work load distribution is different from run to run. Thus the exact run time for each node does not have much meaning and one should focus on the total sum of individual run times.

The last two digits of the total times in Table 5 seem to be erratic. The Unix timing routines do not measure the exact time of a code section only but may include some system times. Thus Table 5 is not erratic.

In order to analyze the performance one should measure the computation time τ_p and communication times τ_m as functions of N_p. Convenient quantities [26] to measure the efficiency are the speedup $S(N_p) \equiv \tau_p(1)/(\tau_p(N_p) + \tau_m(N_p))$ and the efficiency $\epsilon \equiv S/N_p$. In the ideal case of zero communication time we should have $S(N_p) = N_p$ or $\epsilon = 1$. In the above example, communication times are less than 10 milliseconds in all cases. We used the BSD timing routine `cputime` to measure computation times and the MPI routine `MPI_Wtime` for communication times. Since the routine `MPI_Wtime` as implemented in our HP Exemplar machine is not reliable down to seconds, it is not worth reporting. Table 5 indicates that we have almost perfect parallelization, $\epsilon \simeq 1$. Though not listed, we have confirmed it up to $N_p \simeq 32$.

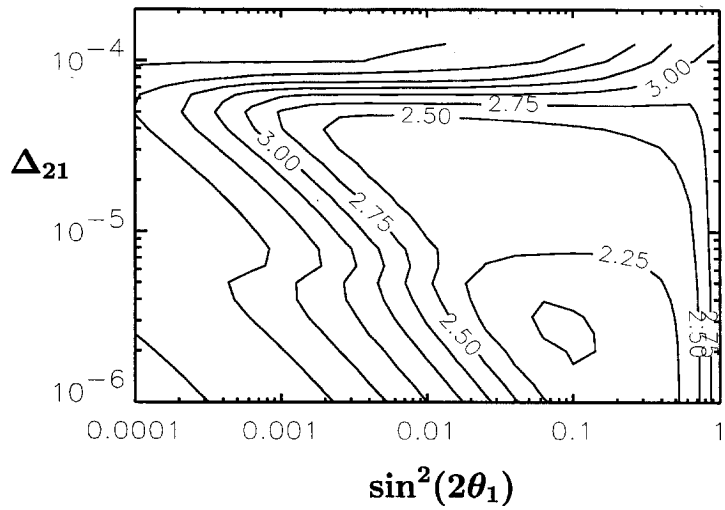

Fig. 4. An iso-SNU plot for B and Be neutrinos with the chlorine detector. Parameter values that were used are ($\Delta_{23} = 2.2 \times 10^{-3}$ eV^2, $\sin^2(2\theta_2) = 0.997$, $\sin^2(2\theta_3) = 0.997$).

5.4. Sample SNU contour plot

With a choice of a parameter set ($\Delta_{23} = 2.2 \times 10^{-3}$ eV^2, $\sin^2(2\theta_3) = 0.997$), we carried out a task for boron neutrinos. Using the Bahcall data for neutrino radial spectrum, energy spectrum, absorption cross sections of the chlorine detector and considering the boron and berillium neutrinos (with $E_\nu = 0.862$ MeV) only, we have made a sample contour plot of SNU (Solar Neutrino Units) in Fig. 4. It seems to be in reasonable agreement with previous results [28,33–45]. For better estimates we should include other neutrinos, increase the grid size, consider the Earth effect and neutrino creation positions off the line of sight, and run more tasks. Detailed comparison with various experimental data will be reported elsewhere.

6. Conclusion

We have successfully overcome difficulties in numerically integrating the evolution equations of the three neutrino species. We have developed an algorithm for computing the survival probability of an electron neutrino in its flight through the solar core experiencing the Mikheyev–Smirnov–Wolfenstein effect. As suggested by Messiah, we adopted a hybrid method that uses an accurate approximation formula in the non-resonance region and numerical integration in the non-adiabatic region. We employed the Bulirsch–Stoer algorithm to integrate the neutrino evolution equation in the non-adiabatic region. The key of our algorithm is to use the importance sampling method for sampling the neutrino creation energy and position and to find the optimum radii to start and stop numerical integration.

We further developed a parallel algorithm for a message passing computer. By setting up a virtual machine where a host and worker nodes reside and letting them exchange job tokens, we have developed a dynamical load balancing mechanism which is effective under any irregular load distributions. Due to the negligible amount of communications and heavy computations, the efficiency was practically one, a perfect parallelization. Thus even though the whole computation of our 21×21 grid takes about 44 days on a single CPU, it will take about one and half day on a 32 node machine. Even if we increase the grid size to 100×100 it will take only about four and half days for the whole computation on a 256 node HP Exemplar X-class machine.

With the choice of parameters, $\Delta_{23} = 2.2 \times 10^{-3}$ eV^2, $\sin^2(2\theta_2) = 0.997$, $\sin^2(2\theta_3) = 0.997$, $E_\nu < 20$ MeV, we have only (1–2) transition. Since most of neutrinos are born near the solar center ($t < 0.15$) where the transition takes place in many cases, our algorithm is still advantageous in this simple case. This advantage will become more prominent when we consider low energy *pp* neutrinos. Supernova neutrinos have to pass through huge and rapidly changing densities of matter and we will see all three transitions. Numerical algorithms will be much more valuable in that case.

Acknowledgements

Both J.S.K. and J.D.K. benefited from discussions with Dr. C.W. Kim while visiting the Korea Institute for Advanced Science (KIAS) in Seoul. We thank Dr. Y.H. Kwon of the the Postech Mathematics Department for giving us valuable advice on numerical algorithms. This work was funded by the Pohang University of Science & Technology.

References

[1] R. Davis Jr., D.S. Harmer, K.C. Hoffman, Phys. Rev. Lett. 20 (1968) 1205.
[2] I. Shelton, IAU Circulars 4316 (24 February 1987).
[3] K.S. Hirata et al., Phys. Rev. Lett. 58 (1987) 1490.
[4] R.M. Bionta et al., Phys. Rev. Lett. 58 (1987) 1494.
[5] E.N. Alexeyev et al., Sov. JETP Lett. 45 (1987) 461.
[6] I. Novikov, Black Holes and the Universe (Cambridge Univ. Press, New York, 1990).
[7] K. Koshiba, Phys. Rep. C 220 (1992) 229;
F. Halzen, Lectures on Neutrino Astronomy, astro-ph/9810368 (1998).
[8] J.N. Bahcall, R.K. Ulrich, Rev. Mod. Phys. 60 (1988) 297;
J.N. Bahcall, M.H. Pinsonneault, Rev. Mod. Phys. 64 (1992) 885; 67 (1995) 781.
[9] S. Turck-Chièze, I. Lopes, Astrophys. J. 408 (1993) 347.
[10] J.N. Bahcall, R. Davis Jr., P. Parker, A. Smirnov, R. Ulrich, Solar Neutrinos (Addison Wesley, Reading, MA, 1994).
[11] C.W. Kim, A. Pevsner, Neutrinos in Physics and Astrophysics (Harwood Academic, Chur, Swiss, 1993).
[12] M. Gell-Mann, P. Ramond, R. Slansky, Supergravity, F. van Nieuwenhuizen, D. Freeman, eds. (North Holland, Amsterdam, 1979) p. 315.
[13] R. Johnson, S. Ranfone, J. Schechter, Phys. Lett. B 179 (1986) 355.
[14] E.J. Chun, C.W. Kim, U.W. Lee, Phys. Rev. D 58 (1998) 093003.
[15] B. Pontecorvo, Zh. Eksp. Teor. Fiz. 34 (1958) 247 [Sov. Phys. JETP 7 (1958) 172].
[16] Y. Fukuda et al., Super-Kamiokande Collaboration, Phys. Rev. Lett. 81 (1998) 1562.
[17] L. Wolfenstein, Phys. Rev. D 17 (1978) 2369.
[18] S.P. Mikheyev, A.Yu. Smirnov, Yad. Fiz. 42 (1985) 1441 [Sov. J. Nucl. Phys. 42 (1985) 913].
[19] L. Landau, Phys. Z. Sowjetunion 2 (1932) 46.
[20] C. Zener, Proc. R. Soc. A 137 (1932) 696.
[21] S.J. Parke, Phys. Rev. Lett. 57 (1986) 1275.
[22] W.C. Haxton, Phys. Rev. D 35, 2352 (1987).
[23] S. Toshev, Phys. Lett. B 196 (1987) 170.
[24] S.T. Petcov, Phys. Lett. B 200 (1988) 373.
[25] C.L. Seitz, Comm. ACM 28 (1985) 22.
[26] G.C. Fox et al., Solving Problems on Concurrent Processors (Prentice-Hall, Englewood Cliffs, NJ, 1988).
[27] K. Hwang, Advanced Computer Architecture (McGraw-Hill, New York, 1993).
[28] T.K. Kuo, J. Pantaleone, Phys. Rev. Lett. 57 (1986) 1805; Rev. Mod. Phys. 61 (1989) 937.
[29] S. Toshev, Phys. Lett. B 198 (1988) 551.
[30] S.T. Petcov, Phys. Lett. B 214 (1988) 139.
[31] W.C. Haxton, W.-M. Zhang, Phys. Rev. D 43 (1991) 2484.
[32] G. Fiorentini, M. Lissia, G. Mezzorani, M. Moretti, D. Vignaud, Phys. Rev. D 49 (1994) 6298.
[33] C.W. Kim, S. Nussinov, W.K. Sze, Phys. Lett. B 184 (1987) 403.
[34] A. Baldini, G.F. Giudice, Phys. Lett. B 186 (1987) 211.

[35] S.P. Mikheyev, A.Yu. Smirnov, Phys. Lett. B 200 (1988) 560.
[36] J.N. Bahcall, W.C. Haxton, Phys. Rev. D 40 (1989) 931.
[37] X. Shi, D.N. Schramm, J.N. Bahcall, Phys. Rev. Lett. 69 (1992) 717.
[38] S.A. Bludman, N. Hata, D.C. Kennedy, P.G. Langacker, Phys. Rev. D 47 (1993) 2220.
[39] D. Harley, T.K. Kuo, Phys. Rev. D 47 (1993) 4059.
[40] L.M. Krauss, E. Gates, M. White, Phys. Lett. B 299 (1993) 94.
[41] P.I. Krastev, S.T. Petcov, Phys. Lett. B 299 (1993) 99.
[42] G.L. Fogli, E. Lisi, D. Montaninos, Phys. Rev. D 49 (1994) 3626.
[43] N. Hata, P. Langacker, Phys. Rev. D 50 (1994) 632.
[44] M. Narayan, M.V.N. Murthy, G. Rajasekaran, S. Uma Sankar, Phys. Rev. D 53 (1996) 2809.
[45] J.N. Bahcall, P.I. Krastev, A.Yu. Smirnov, Phys. Rev. D 58 (1998) 096016.
[46] J.N. Bahcall, Phys. Rev. C 56 (1997) 3391.
[47] J.M. Hammersley, D.C. Handscomb, Monte Carlo Methods (Methuen, London, 1964).
[48] F. James, Rep. Prog. Phys. 43 (1980) 1145.
[49] P.K. MacKeown, Stochastic Simulation in Physics (Springer, Singapore, 1997).
[50] CERN library, http://www.cern.ch/ or asisftp.cern.ch.
[51] R. Bulirsch, J. Stoer, Numer. Math. 8 (1966) 1.
[52] A. Messiah, in: '86 Massive Neutrinos in Astrophysics and in Particle Physics, O. Fackler, J. Tran Thanh Van, eds. (Editions Frontières, Gif-sur-Yvette, 1986) p. 373.
[53] Message Passing Interface, http://www.mpi-forum.org/.
[54] J.S. Kim, Use of asynchronous communications in parallel computing, Proc 2nd International Conf. on Computational Physics, D.Y. Li et al., eds. (Beijing, September 1993) p. 14.
[55] G. Aloisio, G.C. Fox, J.S. Kim, N. Veneziani, IEEE Trans. Signal Processing 39 (1991) 160.